CONCURRENT FIRE DYNAMICS MODELS AND THERMOMECHANICAL ANALYSIS OF STEEL AND CONCRETE STRUCTURES

A Dissertation
Presented to
The Academic Faculty

by

Joonho Choi

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Civil and Environmental Engineering

Georgia Institute of Technology
December 2008

CONCURRENT FIRE DYNAMICS MODELS AND THERMOMECHANICAL ANALYSIS OF STEEL AND CONCRETE STRUCTURES

Approved by:

Dr. Rami Haj-Ali, Advisor
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Kenneth Will
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Donald W. White
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Abdul-Hamid Zureick
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Erian Armanios
School of Aerospace Engineering
Georgia Institute of Technology

ACKNOWLEDGEMENTS

I would like to express my gratitude to all those who gave me the possibility to complete this thesis. Special thanks are due to my advisor Professor Rami M. Haj-Ali whose help, stimulating suggestions and encouragement helped me in all the time of research for and writing of this thesis. I am also deeply indebted to my committee members, Professor Kenneth M. Will, Professor Donald W. White, Professor Abdul-Hamid Zureick, and Professor Erian Armanios.

I want to thank my family and all my friends. It was not possible to complete this thesis without their supports and understanding.

TABLE OF CONTENTS

LIST OF FIGURES

SUMMARY

The objective of this study is to formulate a general 3D material-structural analysis framework for the thermomechanical behavior of steel-concrete structures in a fire environment. The proposed analysis framework consists of three modeling parts: fire dynamics simulation, heat transfer analysis, and a thermomechanical stress analysis of the structure. The first modeling part consists of applying the NIST (National Institute of Standards and Technology) fire dynamics simulator (FDS) where coupled Computational Fluid Dynamics (CFD) with thermodynamics are combined to model the fire progression within the steel-concrete structure. The goal is to generate the spatial-temporal (ST) solution variables (temperature, heat flux) on the surfaces of the structure. The FDS-ST solutions are generated in a discrete numerical form. Continuous FDS-ST approximations are then developed to represent the temperature or heat-flux at any given time or point within the structure. An extensive numerical study is carried out to examine the best ST approximation functions that strike a balance between accuracy and simplicity. The second modeling part consists of a finite-element (FE) transient heat analysis of the structure using the continuous FDS-ST surface variables as prescribed thermal boundary conditions. The third modeling part is a thermomechanical FE structural analysis using both nonlinear material and geometry. The temperature history from the second modeling part is used at all nodal points. The ABAQUS FE code is used with newly developed external user subroutines for the second and third simulation parts. The main objective is to describe the nonlinear temperature-dependency of the specific heat of concrete materials, especially high-strength concretes, that drastically affects their transient thermal solution. New algorithms are also developed to apply the continuous FDS-ST surface nodal boundary conditions in the transient heat FE analysis. The proposed modeling framework is applied to predict the temperature and deflection of the

well-documented Cardington fire tests and to predict the time-to-collapse of the recent Oakland bridge fire caused by a fuel-truck accident.

CHAPTER 1

INTRODUCTION

Structural performance under elevated temperatures from large-scale fire environments can be an important factor that affects the design of structures. The large changes in material behavior of steel under fire lead to rapid changes in the structural geometry due to thermal expansion along with stiffness and strength reduction. Many studies have been proposed over the last decade to model the mechanical behavior of steel structural components and systems under fire. Previous studies on the thermomechanical behavior of structures under fire are reviewed in this chapter. In addition, motivation and goals for the study are described.

1.1 Modeling of the Mechanical Behavior of steel columns and frames under fire

Neves [46] studied the critical temperature of steel columns under fire condition. He calculated the critical temperature of steel columns using a beam element-based program. The effects of the interactions between steel columns and the stiffness of the surrounding structures under fire were considered. Several parametric studies were conducted to examine the factors that influence the critical temperature of the steel columns. The stiffness of the surrounding structures affects the critical value. The thermal elongation of the steel columns is enhanced due to the eccentricity of the load or in the case where the stiffness of the structure was large. The critical temperature was decreased with increasing the stiffness of the surrounding structures and the slenderness of the steel columns. Franssen et al. [21] performed numerical studies of an early Cardington test performed in 1987 using a two-dimensional (2D) steel frame. They considered the heat flow of detailed composite section models with thermal material properties taken from the Eurocode. Good comparison was shown between the predicted

temperature and the test results at the flange, inner and outer web locations for both beam and column sections. Najjar and Burgess [45] developed a nonlinear analysis approach for steel frames under fire with geometric and material nonlinearities. Their program was extensively verified against other numerical results using the BS5950 steel code. They concluded that the code developed can be used for three-dimensional (3D) analysis of multi-story skeletal steel frames under fire. Wang [60] studied two opposite aspects of the effect of structural continuity for steel columns in the steel frame under fire conditions. First aspect was to increase the rotational restraint of the column and then its load carrying capacity is improved. And the other is to increase the column compressive load because of the thermal expansion. From a parametric study, he found out that these two effects almost cancelled each out and a method considering no structural continuity under fire gave reasonable values to calculate the limiting temperature. He also suggested that the fire protection thickness of the column should be increased by more than 20% of the thickness calculated based on column limiting temperature because the column limiting temperature was overestimated for some cases. Valente and Neves [58] forwarded one more step to calculate the critical temperature of the steel columns. They considered the effect of not only axial but also rotational restraints using finite element (FE) program. They found out that the critical temperature of the steel columns was increased with increasing rotational restraint and was decreased with increasing axial restraint. Li and Jiang [40] advocated the use of the Clough force-imbalance method for the analysis of steel frames under fire. They pointed the limitation of the displacement-based FE models and the computational effort that may be required for a cross-sectional analysis. Fire tests were conducted on a small-scale single-story two-bay steel frame and the structural response was successfully simulated. Rodrigues et al. [51] studied an experiment about the compressed steel elements under fire condition. They performed the small-scale tests on compressed steel rectangular bars with thermal elongation elastically restrained. They showed good comparison of the experimental and the

numerical simulation results of the restraining forces according to the temperature of the bar. Ali and O'Connor [5] presented the results for a parametric experimental investigation of the behavior of rotationally restrained steel columns under high temperature. Half scale steel columns were used for the experiments. A method estimating the effective length for fixed end columns under fire conditions was proposed and a comparison with the behavior of a steel column previously tested in fire under axial restraint only was shown. Neves et al. [48] suggested a simple method to calculate the corrected critical temperature of a steel column elastically restrained to the thermal elongation by surrounding structures. A simple steel column model they considered had a rotational and axial spring at the middle of the column and a axial spring at the end of the column. They compared the critical temperatures from their proposed method, Instituto Superior Tecnico (IST) experimental tests and numerical simulation. Their proposed method gave the safer critical temperatures than other methods from the comparison. Vila Real et al. [59] studied the lateral-torsional buckling of steel beams under uniform elevated temperatures due to fire. The aim of the paper is to examine and validate the new and somewhat conservative Eurocode-3 (EN version, part 1-1) and to propose alternatives for the reduction factor to the lateral-torsional buckling load. The beams analyzed were subjected to nonuniform bending. Different residual stress profiles were considered in the response of the steel to high temperatures. The authors used their non-linear FE code that accounts for geometrical and material nonlinearities. The new proposed alternative/method clearly showed improvement over the current design standard.

1.2 Composite steel-concrete structures under fire loads

Concrete materials under elevated temperatures can drastically change their effective mechanical and thermal properties. The increase of temperature can cause phase change in concrete materials during heating, which affects the overall transient heat

conduction and thus the mechanical response within the structural member. Harmathy [28, 29] characterized the thermomechanical material properties of different concrete materials as a function of temperature, such as effective specific heat (thermal capacity), thermal conductivity, mass change rate, and thermal expansion. The ACI 216 report [1] reviewed experimental results characterizing the effect of high temperatures on the effective thermomechanical properties of concrete, e.g. specific heat, coefficients of thermal expansion (CTE), elastic moduli, and stress-strain relationships of normal and light-weight concretes. Design guidance of concrete elements including reinforced concrete slabs and beams, plain and reinforced concrete walls, and reinforce concrete columns under thermal loading, were also discussed. Ahmed and Hurst [2] proposed a thermodynamics-based model that accounted for the gas, water, and solid constituents using conservation of energy in a porous medium. Their 2D model was compared with temperature distributions tests on concrete samples. Their model was used to generate the pore pressure, moisture content, and temperature distribution for high strength concrete columns under fire.

The Eurocode 3 [20] described rules and requirements for the design of steel structures under fire. A set of equations for the steel stress-strain relationship at elevated temperatures was proposed. Thermal elongation of the steel was described including phase change of the crystal microstructure between 750 and 860 °C. Specific heat and thermal conductivity as functions of temperature were also considered. The code provided design equations for tension members, connections, buckling resistance, moment resistance, shear resistance, and lateral-torsional buckling at elevated temperatures. Qutinen and Makelainen [50] performed small-scale tensile tests using the transient state tensile method to study the mechanical properties of structural steels under high temperatures. The objective was to provide accurate temperature dependent mechanical properties for yield strength, elastic modulus, and thermal elongation. Two grades of steel were tested at elevated temperatures. The test results agreed with the

temperature-dependent mechanical properties listed in the Eurocode 3 [20]. Tests were also performed on the steel after cooling down to investigate its residual strength. They concluded that the steel retained its original strength after cooling down. Chen et al. [13] investigated the mechanical properties of high and mild strength structural steels at elevated temperatures. They provided reduction factors over the yield strength and elastic modulus. These factors were compared with four standard codes: American, Australian, British, and European standards. The reduction factors were similar from room temperature up to 540°C. Reduction factors for the yield strengths from the four codes were conservative compared to the test results up to 1000°C. The elastic modulus of high-strength steel from the American, Australian, and European standards were conservative compared to the steady-state tests but unconservative compared to the transient-state tests.

Sanad et al. [53] developed a simple FE model and applied it to analyze the first Cardington fire test. The ABAQUS FE code was used. Two-noded beam elements were applied for the steel beams and a grillage model for concrete slabs. The steel and concrete material properties included nonlinear temperature-dependent stress-strain relationships. Overall predictions were demonstrated for the deflections and temperature profiles at different points in the tested joist. Some of these predictions were in good agreement with the experimental results while others gave large differences. This may indicated that a more refined model is needed, especially highlighting the need for composite shell elements to represent the concrete slab. Gillie et al. [23] developed effective section stress-resultants for reinforced concrete slabs under fire conditions. They developed software called Finite Element Analysis of Shells at high Temperatures (FEAST) to generate the effective cross-sectional response. Compressive and tensile concrete material behaviors as functions of elevated temperatures were considered in the FEAST code. The FEAST consisted of two main modules: Stress-Resultant Analysis of Shells (SRAS) and Finite Element Analysis Interface (FEAI). The SRAS program was

developed to model plates subjected to an arbitrary range of stress-strain-temperature states. The FEAI integrates the SRAS stress-resultants and interfaces with the ABAQUS general purpose FE code. Gillie et al. [24, 25] employed their previous codes and performed structural analyses for the first and third Cardington tests, termed the Cardington British Steel Corner test. Shell elements with eight-nodes and reduced integration were used for the concrete slab while two-node beam element used for the steel beams and columns. However, no fire simulations for the compartments were performed. Instead, a spatially-uniform surface temperature distribution was used for the concrete slab. The temperature history was measured during the Cardington test. Similarly, each beam was subjected to its linear and spatially-uniform cross-sectional temperature profile also measured from the experiment. The fire cooling down phase was not included in these analyses. Their model showed good ability to predict the deflections as function of the prescribed temperature when compared with experimental data from the first and third Cardington fire tests. Lamont et al. [39] developed an adaptive heat transfer program HADAPT used in predicting the temperature within the structural elements in the Cardington tests. The code HADAPT was developed for 2D nonlinear transient thermal analysis and used to generate detailed temperature distributions through the thickness of the ribbed concrete slab and metal deck. Heat convection formulation was used with the interface elements between the concrete plate and metal deck in the composite slab. The formulation allowed for studying parameters, such as density, specific heat temperature dependency, moisture content, convection coefficient, and slab thickness, to find out the temperature changes as a function of these variables. Flint et al. [22] performed numerical analysis to study the mechanical response of a tall building with long span truss floor system under fire environment. They developed a 2D structural model with a multi-floor system using the ABAQUS FE code. A construction type similar to the World Trade Center Towers was used in their computational model. Several floors were assumed to be under a fully flashed-over fire.

The temperature loadings for the fire were applied using a generalized exponential type temperature curve. Heat transfer analysis was carried out for the concrete slab where one side of the slab was heated. Top, bottom, and midpoint temperatures of the concrete slab were reported. They found that while the long truss span floors did not fail, the structure could collapse because the external/perimeter columns failed due to load redistribution. Bailey [8] investigated the structural behavior from the large compartment fire test of the full-scale Cardington building. The fire test of the seven-story concrete building was carried out at the Building Research Establishment (BRE) Laboratories at Cardington, Bedfordshire between July and September 2001. A fire compartment, with a floor area of 225 m^2, was constructed between the ground and the first floor and a number of fire scenarios were considered. Limited test results were reported including residual displacements. The observed damage provided useful information on the holistic behavior of concrete buildings under fire. Significant lateral movement of the heated slab was observed in the tests resulting in buckling of the steel cross-bracing and a lateral displacement of the external columns with extensive spalling to the soffit of the slab. Bailey [8] suggested that the designers should give holistic consideration to the overall structural behavior under fire to prevent premature collapse. Cai et al. [12] used 3D finite element to model general composite beam sections subjected to fire. Their generalized composite beam element included geometrical and material nonlinearity including cracking and crushing of the concrete. Also, temperature-dependent thermal expansion and mechanical degradation of the material were used. Non-uniform temperature profiles, variations of constitutive relationships and thermal strains across the section were considered by dividing the cross section into a number of segments. The proposed element was validated by comparing with theoretical and experimental results. Bailey [9] proposed an extended membrane fire design method to enable the specification of orthotropic reinforcement for the composite slab. The design method gives the designer a tool to specify the most economical planning of reinforcement in the floor slab. A fire

design example for a typical building was performed assuming a 60 minutes duration of fire resistance. He found in the example that the ultimate load-carrying capacity of the floor was increased by 23% by placing more reinforcement in the longer span of the rectangular slab panels. The method was compared with the test results and gave very good correlations. Bailey [10] presented a structural performance-based design approach for steel beams supporting a composite floor allowing designers to specify fire protection to only a proportion of the steel beams within a given floor plate. The new design method enabled an estimate of the membrane capacity of the slab and beam system. This method assumes that the tensile force generated in the beam is transferred to the supporting composite slab and resisted by the compressive membrane force generated around the slab's perimeter. The new design method showed a good match to the BRE corner fire test carried out on the Cardington steel-framed building. Huang et al. [31] presented some comparisons between a simple design method and finite element modeling for composite floor slabs in fire condition. They investigated the influence of thermal curvature, the effect of changing the edge support conditions and the influence of the proportion of steel reinforcement on the structural behavior. They found out that the tensile membrane action can be important in carrying the loads at high temperatures and deflections. Also, the tensile membrane action occurred depended on the aspect ratios of the slabs. The simple design method usually predicts a greater fire resistance than the detailed computer modeling due to tensile membrane action. The membrane action can be a useful tool as a part of a performance-based fire engineering design approach. Elghazouli and Izzuddin [17] examined the behavior of composite steel-concrete building floors under fire conditions. They performed the numerical modeling for the structural behavior in the Cardington fire tests. A grillage model was used for the concrete slabs with geometric and material nonlinearities. Their numerical results were in general agreement with the experimental results. Izzuddin et al. [32] developed a new shell element for concrete composite floor slabs. A geometric orthotropy of the composite

ribbed section was considered in the proposed element using the modified Reissner-Mindlin formulation. Nonlinear concrete material model was included with nonlinear temperature-dependent compression behavior and softening due to cracking in tension. They provided numerical examples comparing their element with the 3D brick element for uniform-thickness and ribbed plates. The new slab element showed good agreement with the 3D-continuum modeling consumed less computational efforts. In a following companion paper by Elghazouli and Izzuddin [18], the new element was used in a full-scale analysis of a composite beam-slab floor system under extreme fire environment present in the third Cardington test. However, coupled fire simulation was not performed. Instead, measured temperature history from the test was used in their full-scale analysis.

1.3 Bridge structures under thermal effect

Limited studies were conducted on the behavior of bridges under combined global and local fire using a realistic fire loading. Dotreppe et al. [15] performed numerical analysis using SAFIR computer code developed at the University of Liege for the Vivegnis Bridge collapse as a result of a past accident. Severe localized fire occurred at one of the bridge foot due to a gas pipe explosion. 3D beam elements were used for main girders, cross girders, concrete slab, arches, and bracings and the suspenders were modeled using truss elements. The model included large displacements and material nonlinearities. The developed model was first performed with room temperature to validate the model comparing with measurements. Then, a transient structure analysis was carried out with increasing temperature. Temperatures were applied to one of the foot of abutment area based on the hydrocarbon temperature-time curve of Eurocode 1, part 1-2. The duration time and failure mode showed good agreement with the observations of the bridge collapse. Mendes et al. [43] performed a 2D partial numerical bridge deck model for Vasco da Gama Bridge under ship fire accident. The cross section temperatures and the safety time, time to collapse, were studied. They used three

different fire designs defined by the geometric characteristics (dimensions, burning rate), the emissive power (thermal radiation, fuel type), and radiation transfer. The effects of convection, thermal irradiation, solar radiation, and fire radiation were also considered for the thermal loading. The predicted temperature distributions of the cross section at several locations were shown with 4 different times (30, 60, 90, and 120 min) for the fire type 3 which was the most severe fire scenario. They decided that the bridge was in the critical status when the temperatures approached 300 ^{o}C because the yield strength of the prestressing steels was decreased around 75% comparing to the room temperature. The anchorage of the bridge was expected to be damaged around 20 to 30 min after fire occurred by their fire-safety criterion.

Several investigations were conducted on bridge behavior due to thermal effect environments including their construction stage. Neves et al. [47] analyzed the effects of wood formwork fire accidents during the bridge construction. The S. Lourenco bridge fire accident was used for this case study. The numerical thermal analysis was also performed for this study using finite element method program. Visual inspection and material testing were helped to decide if the bridge was needed to be repair or replaced. A thermal numerical analysis can be used as support in the decision considering rehabilitation/demolition. Branco and Mendoes [11] proposed a numerical analysis method using the Fourier heat-transfer equation to obtain the nonlinear temperature distributions for the concrete bridges using finite element method. The thermal effects of the local environment conditions were considered in the analysis. The geometry of the cross section, the thermal properties, and the location of the bridge were also taken into account. The behavior of bridges under thermal environment conditions was studied by Moorty and Roeder [44]. The temperature distributions and ranges were obtained using their one-dimensional heat transfer analysis. The movements and stresses of the skew and curved bridges were found out by thermoelastic analysis. The numerical results were verified by a field test. They suggested additional guidelines and recommendations to

accurately predict thermal movements and the placement of bearings and expansion joints. Silveira et al. [56] proposed a numerical method to get the temperature distributions of the concrete bridge using statistical analysis under climatic conditions. The temperature distributions were calculated using two-dimensional Fourier heat-transfer equation which considered the bridge cross section, the thermal properties of concrete and asphalt and climatic conditions. They verified their results with experimental measurements and showed good agreements.

In addition to the above studies on concrete deck bridges, limited studies for fiber reinforced polymer bridge deck systems under thermal gradient loadings, not necessarily due to fire, were also performed. The behavior of Fiber Reinforced Polymer (FRP) bridge deck systems were studied under thermal loading using the linear ABAQUS finite element analysis by Alnahhal et al. [7]. Two types of FRP bridge systems were investigated in this study. One is a hybrid FRP-concrete bridge system and the other is a truss bridge with FRP deck system. The behavior of hybrid bridge system was investigated under thermal loading. The behavior of truss bridge with FRP deck was studied under fire truck thermal and mechanical loading. A simple degradation relationship was applied to consider the temperature dependency of the steel and FRP material. In addition, constant coefficient of thermal expansion, specific heat, and coefficient of thermal conductivity were assumed. Good agreements were shown between FE and experimental strain results. It was important to study the behavior of FRP bridge deck systems considering thermal effects because the coefficient of thermal expansion of FRP was higher than that of concrete or steel. Alnahhal et al. [6] performed numerical simulations to study the behavior of the fiber reinforced polymer (FRP) bridge deck under thermal loading and damage condition using finite element method. Several fire scenarios were carried out in the thermal simulations where burning trucks were under or above the FRP deck bridge. They found out that the elevated temperature was an important factor of the FRP bridge decks and suggested that the damaged area of the

FRP bridge decks be repaired before it was used. However, the material degradation was not considered in their simulations. Also, the damage FRP bridge deck simulations caused by snow and ice plowing process were conducted. They recommended that once the damage was detected, the area must be closed and repaired as soon as possible. The behaviors of glass fiber-reinforced polymer (GFRP) slabs for building and bridge were investigated with experiments under fire conditions by Keller et al. [36]. The ISO 834 fire exposure was used as a fire source and load and non-loaded cases were tested for liquid-cooled cellular GFRP slabs. They found out that the liquid cooling was the effective to improve the fire resistance in the building and the fire endurance of cellular GFRP bridge decks was sufficiently high in the experiments. The critical failure mode of GFRP slabs under fire was not a tensile failure in the damaged hot location but an instability-induced failure on the relatively cold compression side due to the loss of lateral support of the fibers.

1.4 Fire simulation programs

Although the fundamental conservation equations governing fluid dynamics, heat transfer and combustion were first developed over a century ago, the practical mathematical models of fire are relatively recent due to the natural complexity of the problem such as an enormous number of possible fire scenarios to consider and a limitation of computing power. Fire dynamics analysis was developed in some of the studies. The Consolidated model of Fire Growth And Smoke Transport (CFAST) was developed by the Building and Fire Research Laboratory at the National Institute of Standards and Technology [33, 34]. The program is relatively easy to use with low computer requirements and can calculate the time evolving distribution of smoke and fire gasses and the temperature throughout a building during a user-specified fire. CFAST belongs to a zone model code in which each room is divided into two lumped-parameter volumes, an upper layer and a lower layer. The Analysis of Smoke Movement in

Enclosures (JASMINE) has been developed continuously at BRE over 20 years [61]. Scenario can be set-up using the graphical user interface which allows the user to define the geometry and boundary conditions. The users can view the results with a graphical post-processor or commercial Computational Fluid Dynamics (CFD) post processor. The program can simulate fire and smoke movement in 3D for steady state and time-dependent applications. JASMINE is a finite-volume CFD code and is able to model single and multiple compartment enclosures with arbitrary openings, obstructions, fire/heat sources and mechanical ventilation systems. The Simulation of Fires in Enclosures (SOFIE) was developed under the group of a Consortium including a number of European fire research laboratories and initiated at Cranfield University [62]. SOFIE has a text only interface with no graphical pre- or post-processing but is able to export data in either Plot3D or Fieldview format. SOFIE is a CFD code for fire modeling which is applied for determining smoke movement in buildings. The code can predict more complex fire phenomena not normally accessible to general purpose CFD codes such as fire growth and spread, toxic emissions and dispersion, fire-water spray interaction. Fire Risk Evaluation and Cost Assessment Model (FiRECAM) has been developed by the National Research Council (NRC) in collaboration with a number of partners since 1987 [63]. The program can be used to assess the level of fire safety that is provided to occupants in an apartment or office building by a fire safety design. In addition, the model can assess the associated fire costs that include capital and maintenance costs of the fire protection systems and expected fire losses. The user can input the design information for the building through a graphical user interface with pull-down menus. FiRECAM uses statistical data to predict the probability of occurrence of fire scenarios and mathematical models to predict the time-dependent development and spread of a fire, the evacuation of the occupants and the response of the fire department. Fire Dynamics Simulator (FDS) was written by staff members of the Building and Fire Research Laboratory at the National Institutes of Standards and Technology. The program

calculates the temperature, density, pressure, velocity and chemical composition within each numerical grid cell at each discrete time step and computes at solid surfaces the temperature, heat flux, mass loss rate and various other quantities. FDS is a Computational Fluid Dynamics model of fire-driven fluid flow and has a visual post-processing image simulation named “smokeview”.

1.5 Research needs and motivation

Studying the behavior of structures under combined elevated temperatures and mechanical loading is important because of the need to protect civil infrastructure from man-made and natural hazards. Combined fire and mechanical loading can drastically change structural material properties and case large material and geometric deformations. The stiffness and strength of the structure are suddenly decreased and rapid change of initial structural geometry occurs due to thermal expansion.

Many recent studies proposed methods to predict the behavior of the structures under fire. Several structure types such as simple steel columns, steel frames, steel connections, and composite structures are included in these studies. Overall, these proposed simplified coarse computations and are usually limited to structural components or sub-assemblies. Refined 3D coupled transient heat with nonlinear structural analysis methods are needed to better predict the structural damage and collapse. In addition, limited experimental results exist for structural sub-assemblies. These are needed to calibrate and verify modeling approaches. Examples of future research needs for refined analysis of structures under fire are:

- Large scale fire dynamics simulations with multiple fire scenario cases
- Realistic and refined temperatures from a 3D transient heat analysis of structures
- Fully coupled 3D thermal and structural analysis
- Damage models in materials at the structural level
- Analysis of structures under abnormal conditions such as blast prior to fire

- More experimental work to verify the computational models

1.6 Objectives and scope

A concurrent fire-structural modeling framework for composite steel-concrete structures under fire is formulated and examined in this study. Towards that goal, the fire dynamics simulation software (FDS) is used with the ABAQUS FE code for the analysis of the structures under fire. The ABAQUS FE transient heat transfer analysis is first utilized to capture the concrete or steel temperature profiles where the surface heat or temperatures are prescribed from the prior FDS model. A second ABAQUS FE nonlinear 3D stress analysis is performed following the transient heat analysis and is used to simulate the mechanical behavior of the structures. Layered shell and beam elements are both used to model concrete slabs, steel beams, and columns where the temperature is prescribed using results of the fire simulations. The nonlinear multiaxial stress-strain behavior of the steel is recognized using an elastic-plastic formulation where the elastic stiffness and the stain-hardening behavior are temperature dependent.

A new proposed data reduction for the temperature profiles using the fire simulation results is performed using polynomial approximations in temporal and spatial variables. Fire simulations are presented for different fire scenarios, including the analysis of the Cardington tests. Nonlinear large-scale 3D frame structural analyses are carried out with damage assessment. Good overall results are obtained when comparing the analysis with the Cardington tests. The fourth Cardington fire test and the third Cardington fire test are used in this study to validate the proposed analysis framework. The framework is also applied to different steel and concrete bridge structural systems to show the applicability of the proposed analysis framework to predict the behavior of the bridge during fire situations.

CHAPTER 2

PROPOSED CONCURRENT FIRE-STRUCTURAL (CFS) MODELING APPROACH

In this chapter, the FDS fire simulation, a fire load used in the FDS and the method how to combine the FDS simulation results and ABAQUS FE code are described. The FDS can calculate the temperature or heat flux of the surface according to the time. A simple fire load curve is used for the fire source in the FDS. The thermal loads for the steel beams and columns are obtained from the FDS surface temperature profile using a fourth-order polynomial approximation. Also, the thermal loads for the concrete slabs are captured from the heat transfer analysis using same polynomial approximation. The heat transfer analysis is run to get the concrete slab section temperature profile using the FDS surface heat flux results. The thermal stress analysis simulated by the thermal loads from the FDS and heat transfer analysis gives the behavior of structures.

2.1 Engineering models for enclosure fires

There are two categories of numerical fire models. The first includes CFD (computational fluid dynamics) models which are used in a wide range of engineering disciplines. In these models, the considered total volume is divided into lots of sub-volumes and the law of mass, momentum, and energy conservation are performed to the sub-volumes. This modeling methodology is useful when dealing with complex geometries. The other type of fire model is Two-zone model. In this model, the considered compartment is divided into an upper (hot zone) and lower zone (cold zone). Mass and energy conservation equations are solved for both zones for every time step. Some programs using this modeling approach only simulate a fire in a single

compartment and others simulate fires in several compartments linked by doors or ventilation.

2.2 Fire dynamics simulator

The fire dynamics simulator (FDS) has been developed at the Building and Fire Research Laboratory (BFRL) at the National Institutes of Standards and Technology (NIST), e.g. McGrattan et al. [42]. The program calculates the temperature, density, pressure, velocity, and chemical composition within each numerical grid cell at each discrete time step. It computes the temperature, heat flux, and mass loss rate of the enclosed solid surfaces. The FDS code is formulated based on Computational Fluid Dynamics (CFD) of fire-driven fluid flow. The FDS numerical solution can be carried out using either a Direct Numerical Simulation (DNS) method or Large Eddy Simulation (LES). The latter is relatively low Reynolds numbers and is not severely limited in grid size and time step as the DNS method. In addition to the classical conservation equations considered in FDS, including mass species momentum and energy, thermodynamics-based state equation of a perfect gas is adopted along with chemical combustion reaction for a library of different fuel sources. The latter is used in the case where the fire heat release rate is unknown. FDS also has a visual post-processing image simulation program named "smokeview." This study develops different software for post-processing the FDS results and generating the temporal and spatial numerical data needed for the proposed temperature approximation functions.

2.3 Prescribed fire loads

The surface energy or heat release rate (HRR) per unit area of a fire source can be prescribed and numerically characterized directly avoiding calculating the heat release using chemical combustion reaction. A relatively simple and practical form of the HRR curve is described in Karlsson and Quintiere [35] and shown in Figure 2-1. The curve is

divided into three parts: the growth phase, the steady-state phase, and the decay phase. The initial fire development is accelerating in the first part and can be described using a quadratic function of time. The second phase is a steady-state phase with a magnitude Qmax. The duration of the steady phase is determined by some assumptions on the fire scenario and content of the fire load. The fire decay phase is also dependent on the scenario and content. Without detailed knowledge, this study assumes the decay function as a reflection of the growth phase function. Karlson and Quintiere [35] provides tables for typical coefficients that can describe different HRR coefficients depending on the items consumed in the fire, such as wood, sofa, electronics, etc.

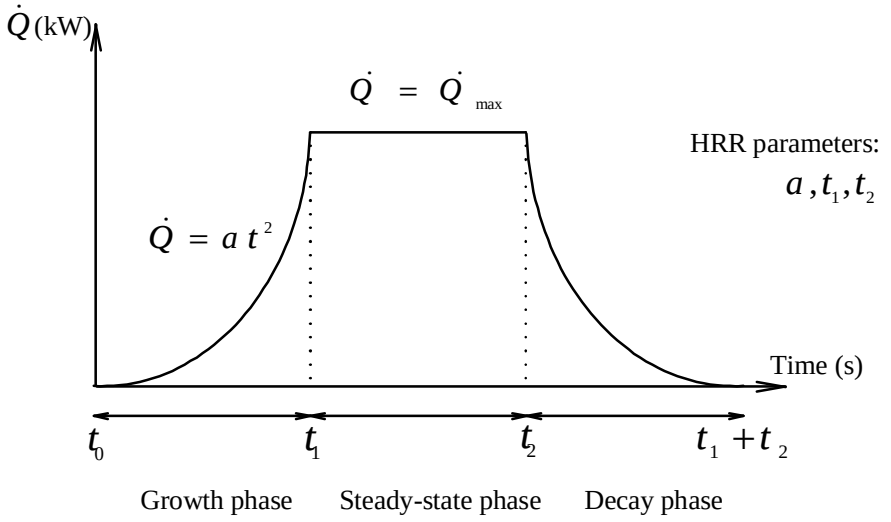


Figure 2-1. A simple fire curve

2.4 Proposed Concurrent Fire-Structural (CFS) Modeling

In this modeling approach, analysis of structures under combined mechanical and fire loads are carried out using the FDS and ABAQUS FE codes. The FDS is first used to solve for the temperature or heat flux on the surfaces as a function of time. A simple heat release rate as a function of time is used to represent the energy released from the surface of the fire source in the FDS model. The solutions for the surface heat and temperature of the steel beams and columns obtained from the FDS fire model are

approximated using a fourth-order polynomial in time with coefficients that are spatial dependant. The temperature distributions within the steel beams are assumed to be linear as a function of the upper and lower surface temperatures. However, the 3D temperature distribution for the concrete slabs is more complicated and is determined from a separate heat transfer analysis using the surface heat flux polynomial approximations as boundary conditions. The nonlinear structural analysis is sequentially carried out once the temperature distributions are obtained as a function of time within all points in the structure.

Figure 2-2 shows a general framework for the analysis of structures under combined fire and mechanical loading. The proposed approach can be divided into three simulation parts. The first part is the fire simulation where the FDS model is utilized. The FDS model generates a solution of several state variables, such as pressure, temperature, heat, velocity vector. However, our framework is interested in the heat and temperature solution part that are related to the structure performance and response. The temperature and heat flux of the interior structural surfaces profiles are used and applied to subsequent simulation parts. The second part is the heat transfer analysis. The objective of this part is to compute the temperature profiles for the different structural components, beams, columns, and especially the concrete slabs through their thickness using the heat flux or surface temperature results from the FDS model. The third part is the nonlinear 3D structural analysis. The temperature profiles in this modeling stage is known and imposed spatially as a function of time.

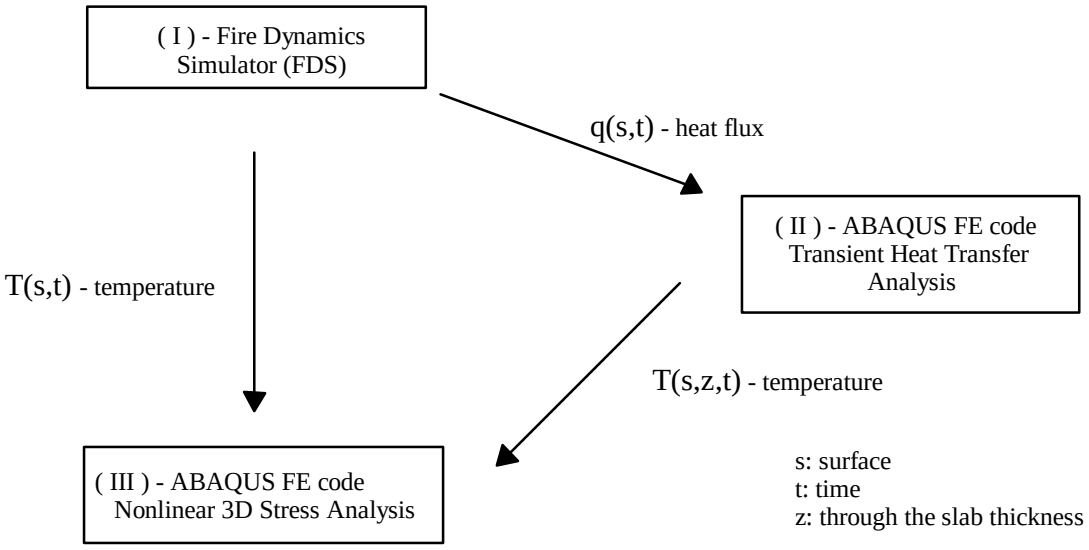


Figure 2-2. Proposed concurrent fire-structural modeling framework

CHAPTER 3

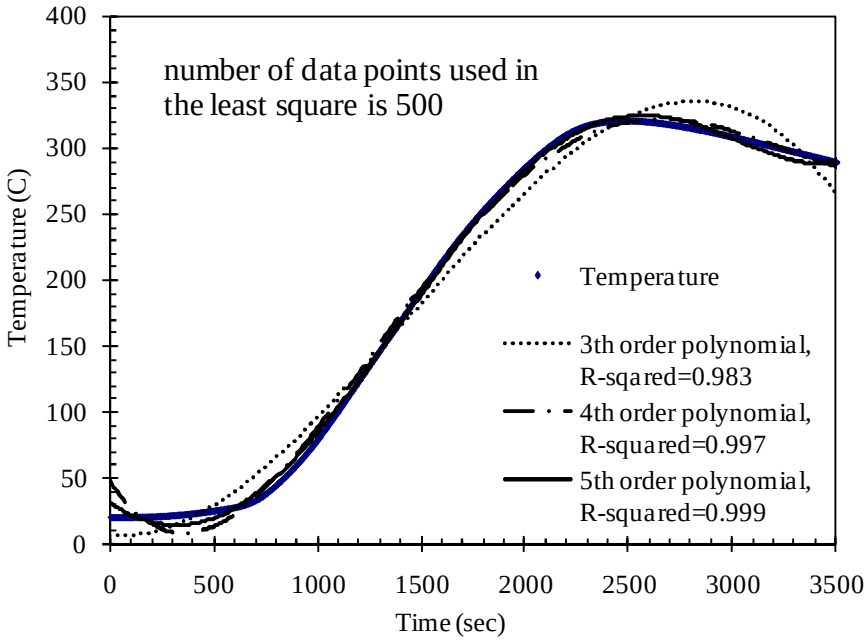
SPATIAL – TEMPORAL TEMPERATURE APPROXIMATIONS IN STRUCTURAL ELEMENTS UNDER FIRE

Spatial – temporal temperature approximations are newly developed in this Chapter in order to be used in the proposed concurrent fire – structural modeling approach. These approximations are applied using the results from the fire simulations and imposed in the finite element structural analyses. The main idea is to reduce the huge amount of data from fire simulations as a function of time to continuous spatial-temporal polynomial equations form.

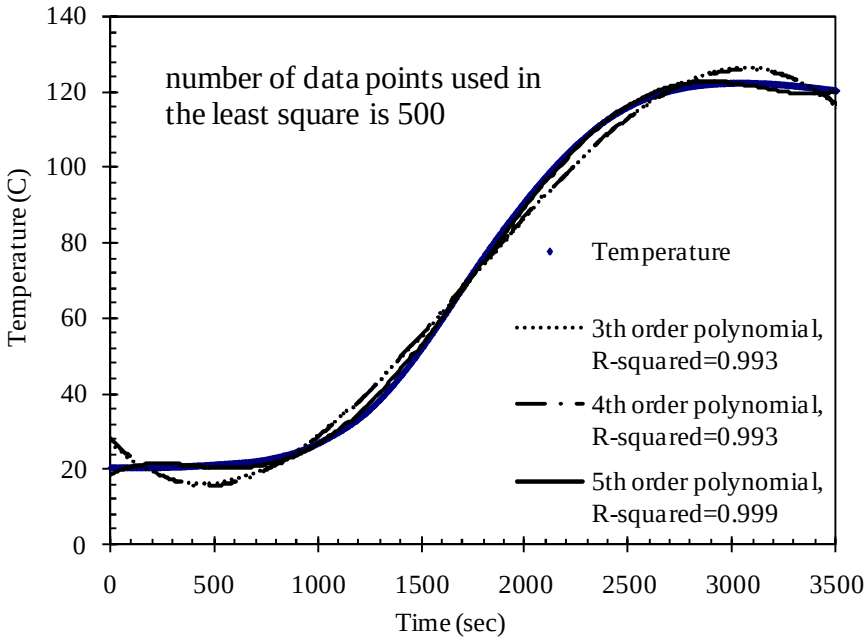
3.1 Spatial-temporal temperature approximation equation

The interior surface boundary file which is one of the Fire Dynamics Simulator (FDS) result files contains the all interior surface temperature or heat flux values during the fire simulation. The large volume of boundary data generated in a typical fire simulation makes it difficult, if not impossible, to use directly in the subsequent structural analysis. Methods for data reduction and simplified fire-simulation results must be used in order to effectively simulate the refined temperature distribution and at the same time retain the efficiency of applying the temperature profiles. Polynomial regression type equations are used to present a simplified form of the temperature or heat flux distributions. Polynomial equations with 3^{rd}, 4^{th}, and 5^{th} orders are examined to decide the efficient order for time and space suitable for large-scale structural analyses. Figure 3-1 shows the FDS temperature results with different order polynomial equations as a function of time. The polynomial equations are obtained using a least square type error minimization and five hundred points are used in this method. The R-squared values of the 3^{rd}, 4^{th}, and 5^{th} are 0.983, 0.997, and 0.999, respectively in Figure 3-1(a). In addition,

another beam element temperature is compared with several order equations in Figure 3-1(b). It shows 0.993, 0.993, and 0.999 as the R-squared values for 3^{rd}, 4^{th}, and 5^{th} order polynomial. The 5^{th} order polynomial equation is chosen to represent the temperature through time based on these temperature comparisons. The order of polynomial equation for spatial distribution is determined in the same manner. Temperatures and different order polynomial equation results are shown in Figure 3-2 as a function of distance. The number of points used in these are twenty five. The spatial temperature distribution shows more even graph than temporal temperature distribution. The R-squared values of 3^{rd}, 4^{th}, and 5^{th} are calculated as 0.982, 0.998, and 0.998, respectively for a beam C (Figure 3-2(a)). 0.987, 0.990, and 0.993 of R-squared values are also obtained for other beam case shown in Figure 3-2(b). The 4^{th} order polynomial is determined to represent the temperatures for spatial distribution from the results. There could be other higher order good approximation equations to express the temperature profile, but in this study the 5^{th} and 4^{th} order polynomial equation are chosen for the spatial – temporal approximation because they strike a good balance between accuracy and simplicity. The comparison between FDS results and the proposed approximation are shown in the next section.

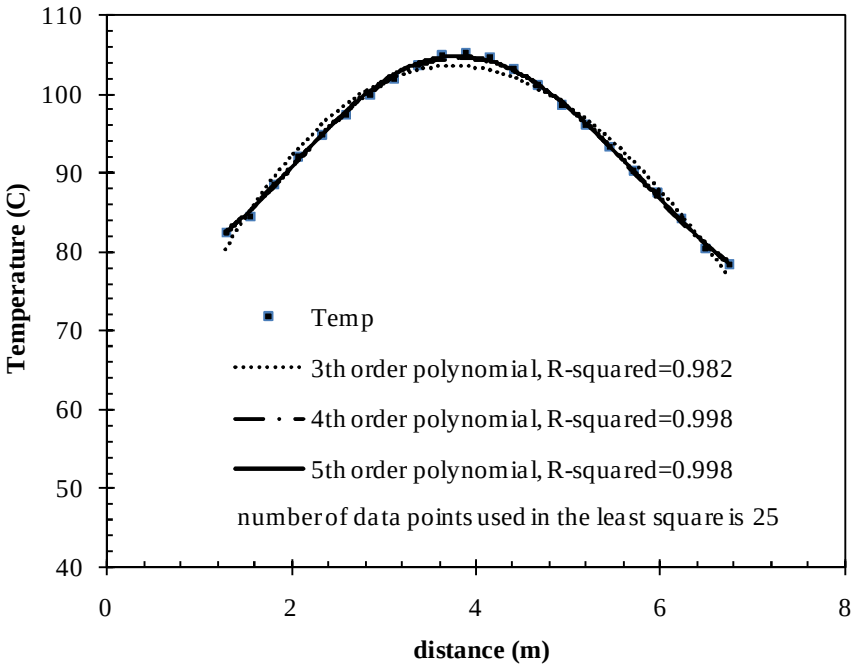


(a) Beam A

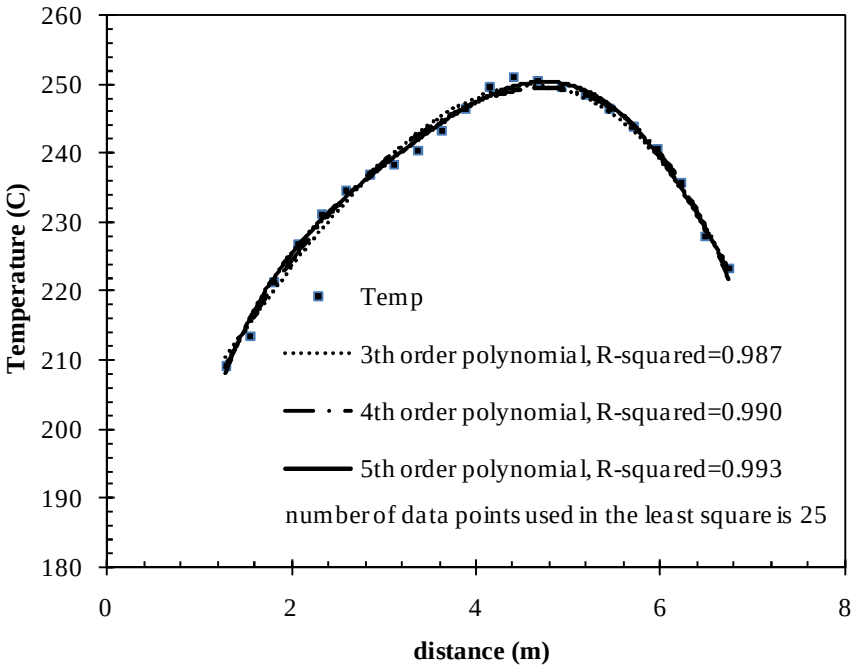


(b) Beam B

Figure 3-1. Temperatures with different order polynomial equations through time. Five hundred data points are used.



(a) Beam C



(b) Beam D

Figure 3-2. Temperature with different polynomial equations through distance. Twenty five data points are used.

3.2 Surface temperature history representations

This study employs a unified approximation for the temperature in beams and columns as a fourth-order polynomial in terms of the structural axial coordinates. The five coefficients of each polynomial are time dependent functions, each in the form of a fifth-order polynomial in time. This can be described for a one-dimensional (1D) element in the s-direction (Figure 3-5(a)) as:

$$T(s) = A_0(t) + A_1(t)s + A_2(t)s^2 + A_3(t)s^3 + A_4(t)s^4$$
$$A_0(t) = A_{00} + A_{01}t + A_{02}t^2 + A_{03}t^3 + A_{04}t^4 + A_{05}t^5$$
$$A_1(t) = A_{10} + A_{11}t + A_{12}t^2 + A_{13}t^3 + A_{14}t^4 + A_{15}t^5$$
$$A_2(t) = A_{20} + A_{21}t + A_{22}t^2 + A_{23}t^3 + A_{24}t^4 + A_{25}t^5$$
$$A_3(t) = A_{30} + A_{31}t + A_{32}t^2 + A_{33}t^3 + A_{34}t^4 + A_{35}t^5$$
$$A_4(t) = A_{40} + A_{41}t + A_{42}t^2 + A_{43}t^3 + A_{44}t^4 + A_{45}t^5$$

where the $A_0(t)$, $A_1(t)$, $A_2(t)$, $A_3(t)$, and $A_4(t)$ are time-dependent polynomials (coefficients). A least square type error minimization is carried out to obtain the best polynomial coefficients that minimize the overall error. Once the approximation process is complete, the result is in the form of a coefficients' matrix (5 by 6) for each axial structural element. The total of thirty coefficients is sufficient to provide for the element's spatial and temporal (s,t) temperature during the fire. Figure 3-3 and 3-4 illustrate the proposed temperature approximation compared with the actual FDS simulation results for beam and column type elements and in different fire scenarios. Twenty one and five hundred points are used to determine the polynomial parameters for Figure 3-3 and Figure 3-4, respectively. Overall, the proposed polynomial form and its order, in time and spatial variables, strikes an acceptable balance between simplicity and accuracy. This minimal computational storage premium is minute compared to the large numerical database generated during the analysis for a typical member at its grid points. The same fourth-order polynomial approximation scheme is used to represent the heat and temperature for an interior surface, such as for concrete slabs. In this approach, the surface is divided into several lines where the temperature or heat polynomial

approximation in (s,t) are carried out independently. Each of them has a fourth-order polynomial approximation equation for the interior surface heat flux. Figure 3-5(b) shows how to divide the slab to axial parts.

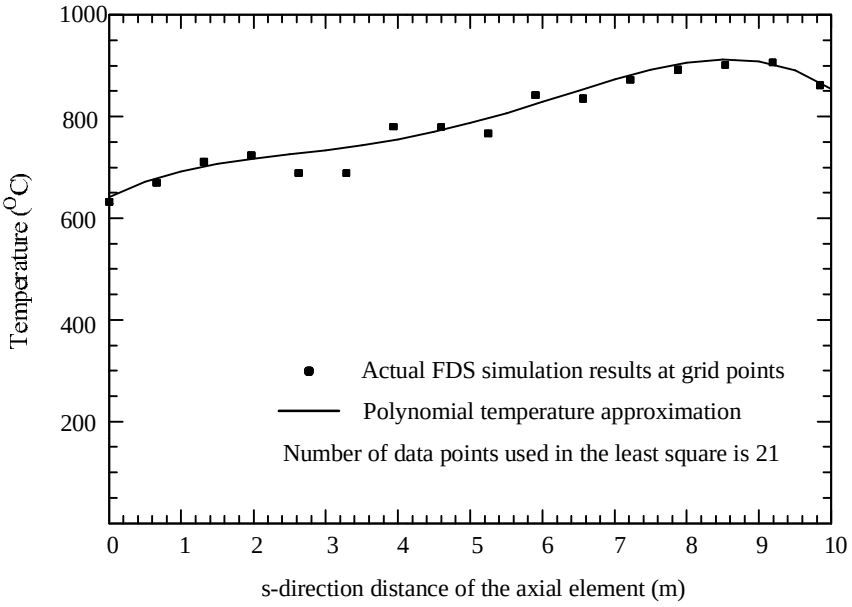


Figure 3-3. Temperature comparison between FDS results and polynomial approximation at a certain time. The ST polynomial parameters are determined based on least square method with 21 sampling data points.

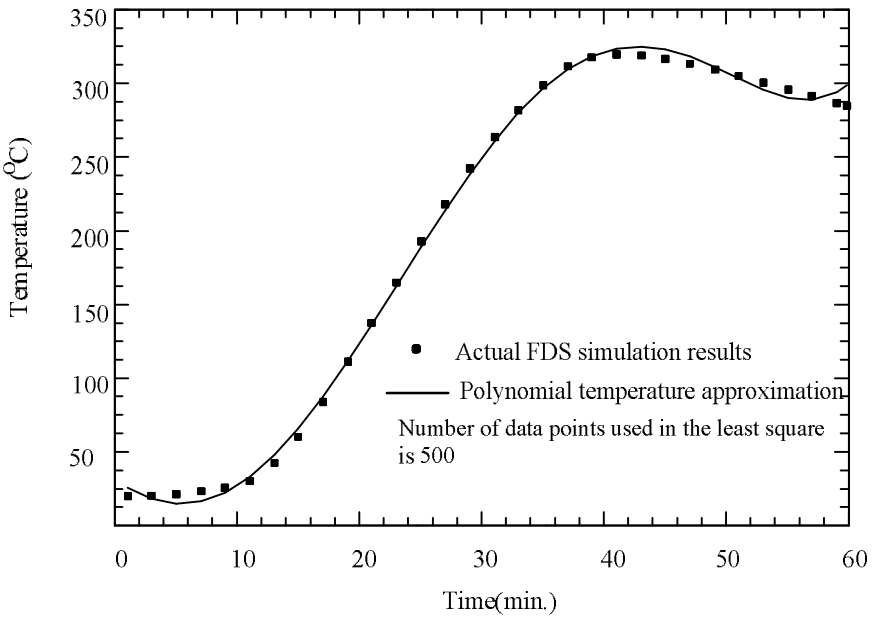
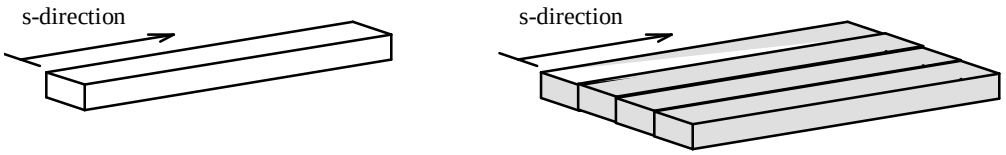


Figure 3-4. Temperature comparison between FDS results and polynomial approximation at a certain location. The ST polynomial parameters are determined based on least square method with 500 sampling data points.



(a) An axial part of a beam or column (b) Axial parts of a slab

Figure 3-5. Axial structural parts for fourth-order polynomial approximation

CHAPTER 4

NONLINEAR HEAT TRANSFER FORMULATION AND MATERIAL PROPERTIES UNDER ELEVATED TEMPERATURES

In this chapter, the governing 3D heat conduction equations for concrete materials are derived. The material properties in the heat conduction equations, such as effective specific heat (C_{eff}) and conductivity (k) are assumed to be dependent on the temperature and material contents. Temperature dependence and degradation of some mechanical properties are taken into account in the mechanics formulation, such as elastic moduli, thermal expansion coefficients, and nonlinear stress-strain relations as a function of temperature. Heat equations derivations and temperature-dependent thermal and mechanical material properties are presented in this chapter.

4.1 Nonlinear heat conduction in solids

The energy conservation principle can be stated as the internal heat energy rate stored inside a solid is equal to the balance of externally supplied and emitted heat energy fluxes transmitted on the surface of the solid. This is simply expressed as:

$$\dot{\overline{Q}}_{in} = \dot{\overline{Q}}_{out} \tag{4-1}$$

The internal volumetric heat energy rate produced inside the solid, $\dot{\overline{Q}}_{in}$, can be rewritten as :

$$\dot{\overline{Q}}_{in} = \int_V \frac{\partial}{\partial t}(\rho Q_{in})dV = \int_V \left(\rho \dot{Q}_{in} + \dot{\rho} Q_{in}\right)dV \tag{4-2}$$

where, $\overline{Q}_{in}$ is the internal heat energy density per unit mass, and is the density rate. The rate of the internal energy density can be rewritten using the derivative chain rule as:

$$\frac{dQ_{in}}{dt} = \frac{dQ_{in}}{dq(x,t)}\frac{d\theta(x,t)}{dt} = C_p(q)\frac{dq(x,t)}{dt} \tag{4-3}$$

where $q(x,t)$ is the current spatial temperature field, and x denotes the spatial location vector in 3D space. The rate of the internal energy density with respect to the temperature expresses the internal heat capacity absorption property of the material, known as the specific heat capacity (C_p). Substituting Eq. (4-3) into Eq. (4-2), and assuming a constant density, the rate of the internal energy density yields:

$$\dot{\overline{Q}}_{in} = \int_V \rho\frac{dQ}{dt}dV = \int_V \rho\frac{dQ_{in}}{d\theta(x,t)}\frac{d\theta(x,t)}{t}dV = \int_V \rho C_p(\theta)\frac{d\theta(x,t)}{t}dV \tag{4-4}$$

Fourier's law states that the heat flux rate transferred through surface is proportional to the heat conductivity times the spatial gradient of the temperature. Therefore, the total external energy rate can be obtained from the surface integral:

$$\dot{\overline{Q}}_{out} = \int_s q_i \cdot n_i\, ds = \int_s \left(k(\theta)\frac{d\theta(x,t)}{dx_i}\right)\cdot n_i\, ds = \int_V \frac{d}{dx_i}\cdot\left(k(\theta)\frac{d\theta(x,t)}{dx_i}\right)dV \tag{4-5}$$

where q_i is the surface heat flux vector, and k is the thermal conductivity. Combining Eq. (4-4) and (4-5), the partial differential (heat conduction) equation (PDE) can be developed as:

$$\int_V \rho C_p(\theta)\frac{d\theta(x,t)}{dt}dV = \int_V \frac{d}{dx_i}\cdot\left(k(\theta)\frac{d\theta(x,t)}{dx_i}\right)dV \tag{4-6}$$

Since the volume is arbitrary, Eq. (4-6) becomes:

$$\rho C_p(\theta)\frac{d\theta(x,t)}{dt} = \frac{d}{dx_i}\cdot\left(k(\theta)\frac{d\theta(x,t)}{dx_i}\right) \tag{4-7}$$

Eq. (4-7) can be considered as linear PDE, if ρ, C_p, k are constant and solved using classical numerical techniques, including the finite element (FE) method. In the case where the latter material properties are functions of temperature, the PDE is a nonlinear and a more complex numerical technique is required for general solutions, e.g. concrete under high temperatures. In this study, the ABAQUS general purpose FE code is used

for both the nonlinear transient heat and the thermomechanical analyses. In the former case, the concrete temperature dependent specific heat capacity is included as an external code used to describe the material thermal behavior. This general code basically updates the internal thermal energy per unit mass at the end of the time increment. The temperature gradient of the internal energy is also required as well as the matrix form of the gradient spatial derivatives. Finally, the heat flux and its gradients are also required to complete the nonlinear heat transient analysis. In our code, we define the temperature-dependent heat capacity and conductivity as internal flux terms calculated as part of the total internal energy density.

4.2 Temperature-dependent mechanical and thermal properties of concretes

Engineering concrete material properties are non-uniform under elevated temperature. The thermal and mechanical material properties of concrete, such as specific heat, conductivity, CTE, elastic modulus, and stress-strain relationship, are temperature-dependent especially at relatively high temperatures.

This section describes needed concrete material properties and relevant tests characterizations under elevated temperatures. Figure 4-1 describes the effective specific heat (thermal capacity), C_{eff}, of normal strength concrete (NSC) at a wide range of temperatures. The effective specific heat is obtained as specific heat times the current density of concrete. Lower and upper bounds are identified based on Harmathy [28,29] where several other reported results for different concretes are within these bounds depending on the aggregates, admixtures, and mix proportions. Kodur and Sultan [38] performed experiments on the thermal properties of high strength concrete (HSC) based on different aggregates type and show the composition of aggregates effect the thermal behavior of the concrete. Figure 4-2 shows the effective specific heat of HSC as a function of temperature for both siliceous and carbonate based aggregates.

The conductivity of NSC and HSC is also temperature-dependent and both decrease as temperature increases. Figure 4-3 shows experimental values of conductivity of NSC in the form of upper and lower bound as a function of temperature while Figure 4-4 describes the conductivity as a function of temperature for HSC materials with siliceous and carbonate based aggregates. The conductivity of HSC at room temperature is between the identified upper and lower bounds of the conductivity of NSC materials, however, the conductivity of HSC decreases more rapidly than NSC. The carbonate based HSC is most sensitive to temperature increase and it decreases to about 12.5% of initial value at 1000°C. This means that the material absorbs more energy, so, more time is needed to disperse the temperature field.

Concrete compression and tension stress-strain curves used in this study are shown in Figure 4-5 and 4-7, respectively. While the compression behaviors are provided from the Eurocode-2, tension curves are not provided at the elevated temperatures. Therefore the tension stress-strain curves shown in Figure 4-7 for elevated temperatures are developed based on room temperature experimental results reported by Gopalaratnam and Shah [27] shown in Figure 4-6. The experimental results are re-constructed such that the slopes in both tension and compression match. The new curve (solid line in Figure 4-6) is taken and scaled based on the stress degradation ratios obtained from the compression stress-strain curves as shown in Figure 4-6. For example, to construct the tension stress-strain behavior for uniform 100°C, the slope of the room-temperature (RT) tensile curve is multiplied by the compression slope degradation ratio (from RT to 100°C). The ultimate stress for the 100°C curve is also found by multiplying the RT ultimate stress with the corresponding compression ratio. The post-ultimate softening behavior curves are re-constructed in the same fashion using the provided RT tension softening. Experimental work for the thermal and mechanical material properties of HSC, performed by Cheng et al. [14] and Kodur et al. [38] is also used in this study. Similarly, the tension stress-strain relations at elevated temperatures

for HSC are scaled from the room temperature data reported by Marzouk and Chen [68]. Figure 4-9 illustrates the assumed tension stress-strain curves for HSC at elevated temperatures using room temperature curves but scaled based on Figure 4-8.

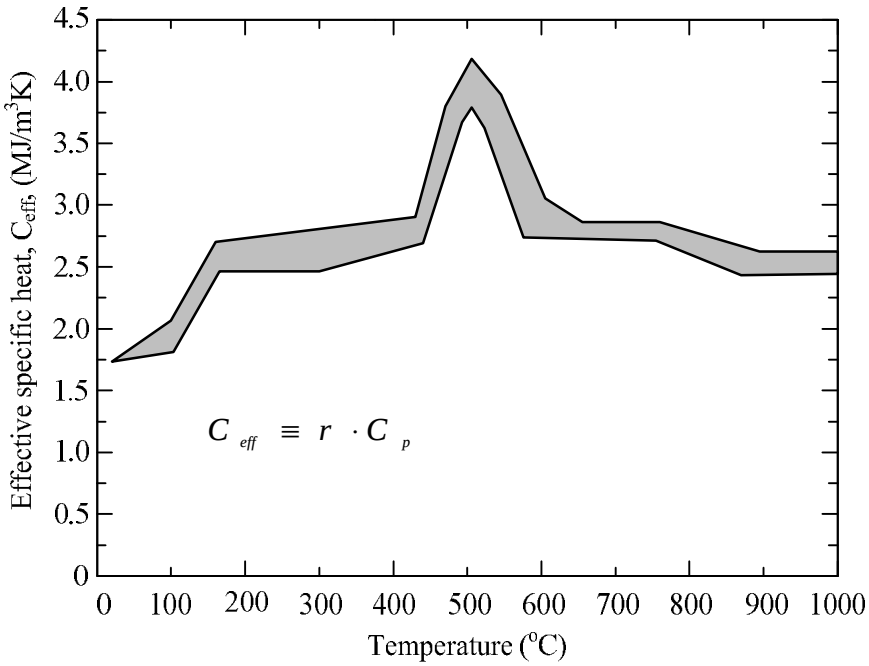


Figure 4-1. Upper and lower bounds of effective specific heat test results of NSC as a function of temperature adapted from Harmathy [28,29]; Different tests values are within the bounds depending on the aggregates, admixtures, and mix proportions.

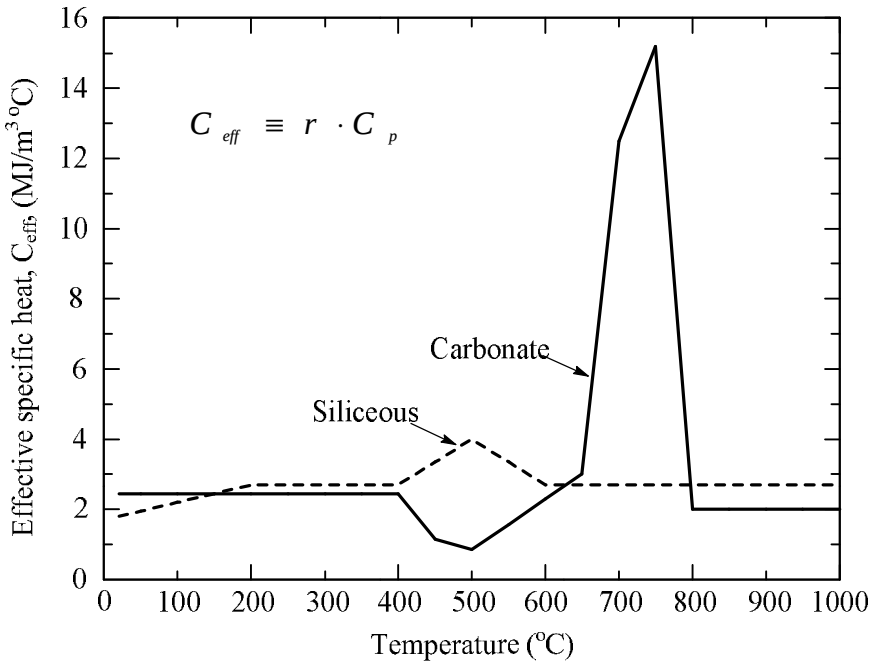


Figure 4-2. Effective specific heat of siliceous and carbonate aggregates based HSC as a function of temperature.

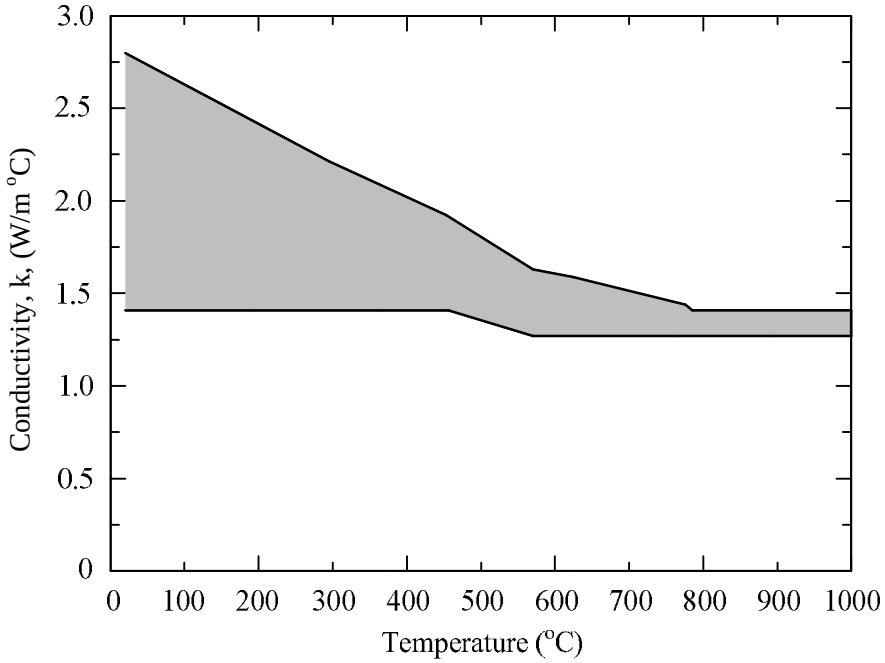


Figure 4-3. Upper and lower bounds of conductivity of NSC as a function of temperature adapted from Harmathy [28,29].

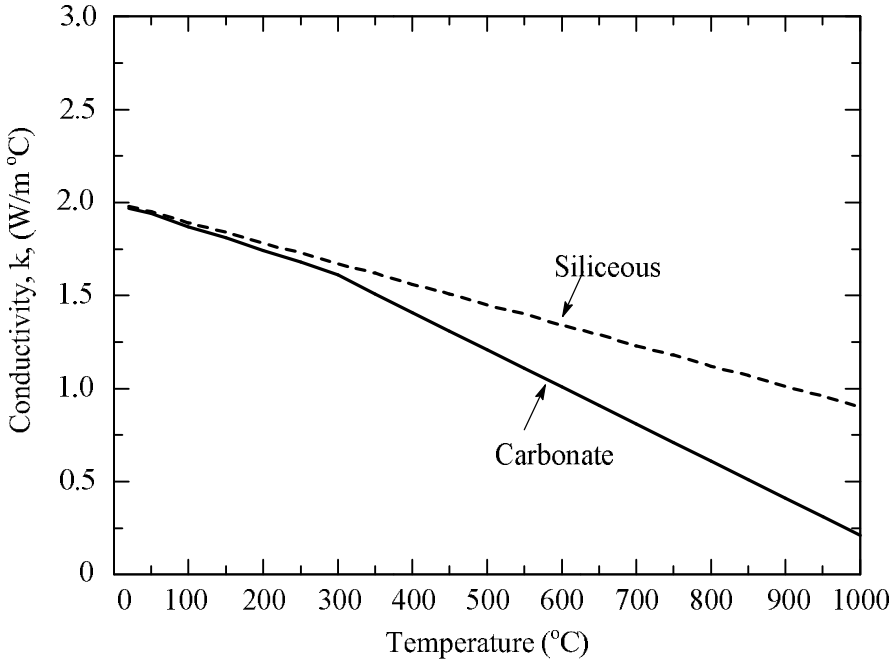


Figure 4-4. Conductivity of siliceous and carbonated aggregate based HSC as a function of temperature.

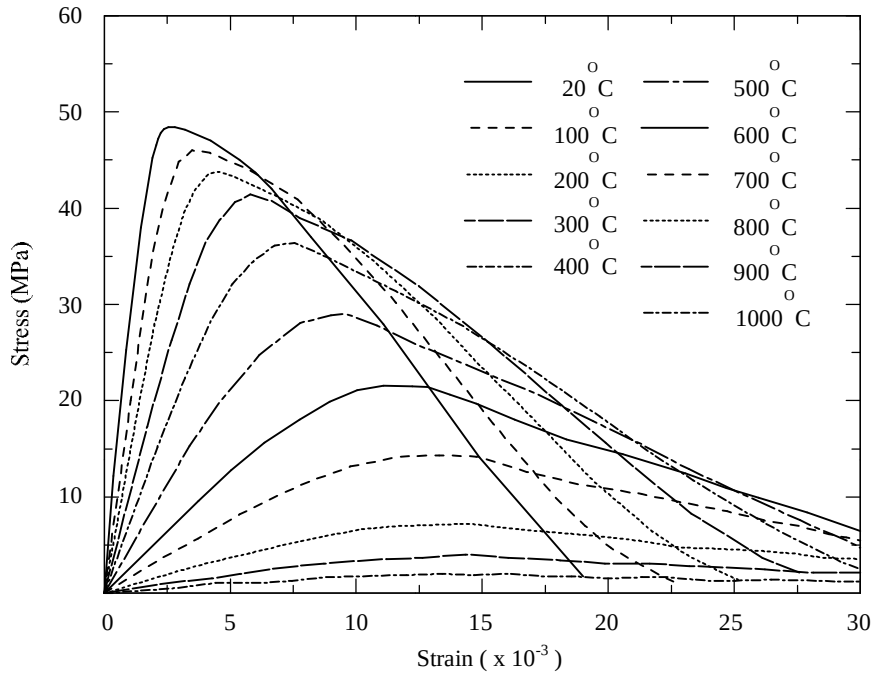


Figure 4-5. Normal strength concrete compression stress-strain curves at elevated temperatures. (Adapted from Eurocode-2, ENV 1992-1-2:1995[19])

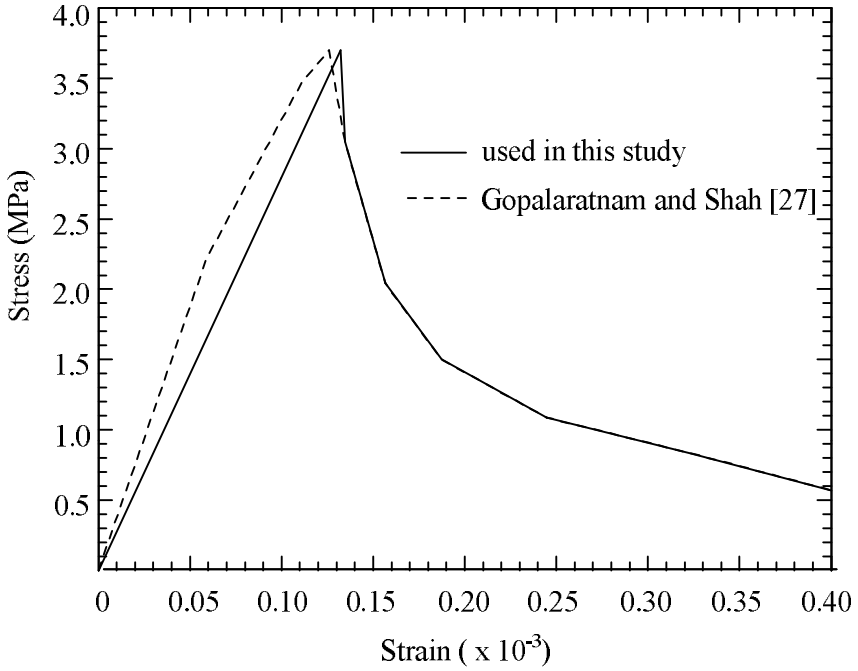


Figure 4-6. Normal strength concrete stress-strain relationships in tension at room temperature used in this study and experimental data

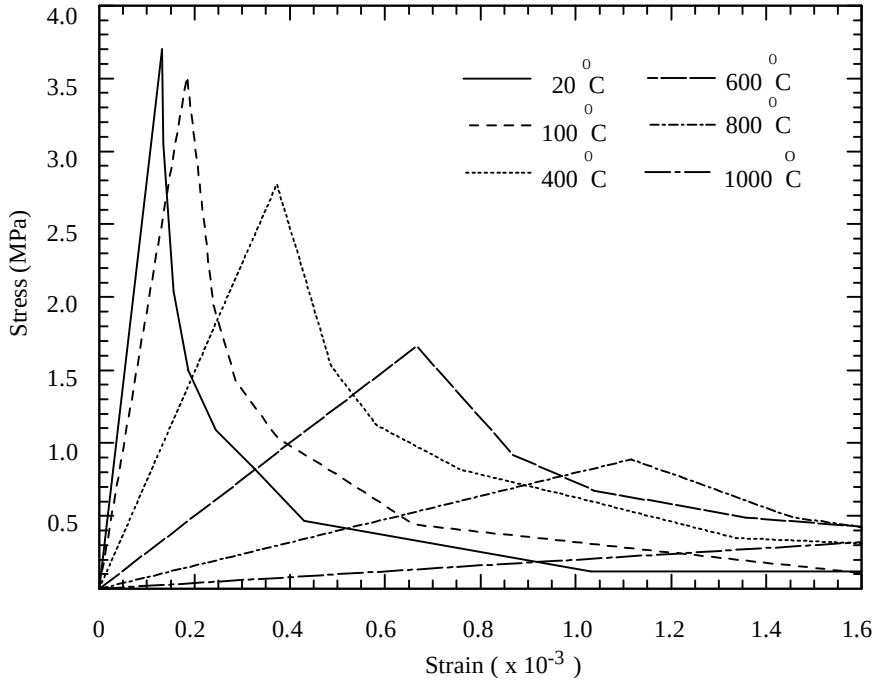


Figure 4-7. Re-constructed normal strength concrete tension stress-strain curves at elevated temperatures

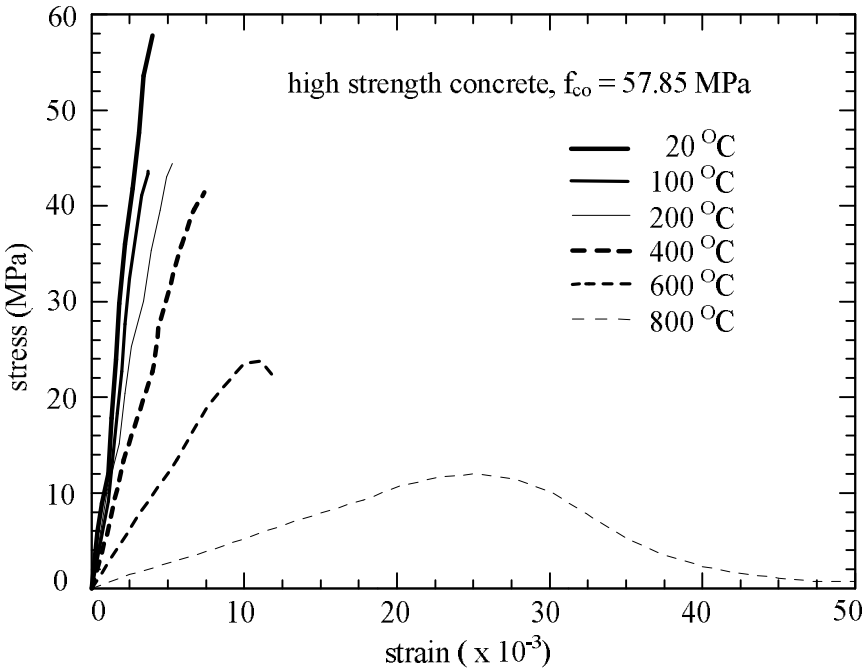


Figure 4-8. High strength concrete compression stress-strain curves at elevated temperatures (Adapted from Cheng et al. [14])

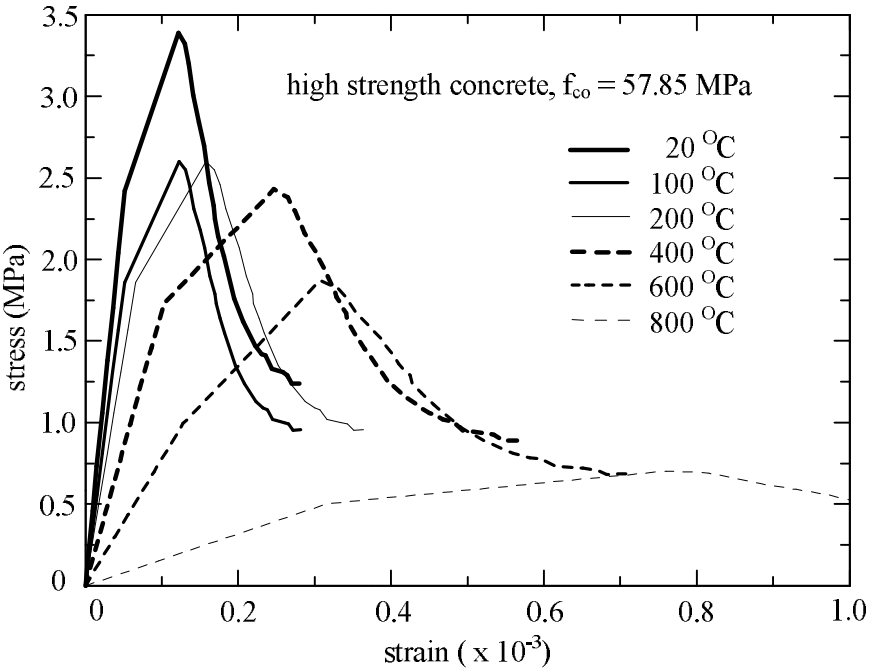


Figure 4-9. Re-constructed tension stress-strain curves of high strength concrete at elevated temperatures

4.3 Material properties of steel under elevated temperature

The mechanical properties of steel and concrete under elevated temperatures are needed in the thermomechanical analysis. The mechanical degradation of these construction materials and their thermal expansion as a function of temperature is needed in order to properly carry out the proposed nonlinear stress analysis. The material in steel beams and columns can be modeled using incremental elasto-plastic constitutive models. Degradation of the stress-strain response as a function of increasing temperatures should be incorporated with the nonlinear material models. Figure 4-10 shows the axial stress-strain curves for steel under uniform temperatures. The coefficient of thermal expansion (CTE) for steel is not constant but increases as temperature increases. The CTE for steel can be calculated from Eq. (4-8) proposed by the ACI Committee 216 [1] as:

$$a_{CTE} = (11 + 0.0036 \times T) \times 10^{-6} [1/^{\circ}C] \tag{4-8}$$

where, T is temperature in Celsius degrees.

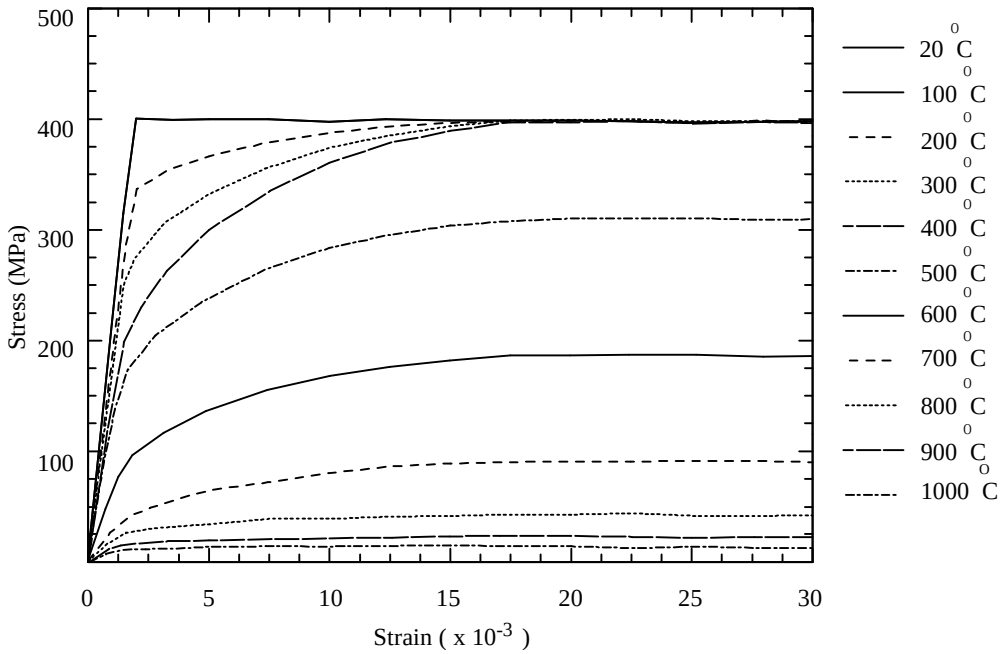


Figure 4-10. Steel uniaxial stress-strain curves at elevated temperatures. (Adapted from Eurocode-3, ENV 1993-1-2:1995 [20])

4.4 Nonlinear temperature-dependent steel constitutive model

A classical metal plasticity model which uses Mises yield surfaces for isotropic yield and isotropic hardening behavior is used in this study for the steel material. The Mises yield surfaces assume that yield of the metal is independent of the equivalent pressure stress. The Mises yield surface is defined by giving the value of the uniaxial yield stress as a function of uniaxial equivalent plastic strain and temperature. Isotropic hardening means that the yield surface changes size uniformly in all directions such that the yield stress increases (or decreases) in all stress directions as plastic straining occurs.

The yield stress can be given as a tabular function of plastic strain of temperature in ABAQUS FE. The yield stress at a given state is simply interpolated from this table of data, and it remains constant for plastic strains exceeding the last value given as tabular data.

4.5 Inelastic strain-softening temperature-dependent constitutive model

There are three different constitutive models for the analysis of concrete provided in ABAQUS: the smeared crack concrete model, the brittle cracking model, and the concrete damaged plasticity model. Each model is designed to provide a general capability for modeling plain and reinforced concrete. The smeared crack model is intended for applications in which the concrete is subjected to essentially monotonic straining and a material point exhibits either tensile cracking or compressive crushing. Plastic straining in compression is controlled by a "compression" yield surface. Cracking is assumed to be the most important aspect of the behavior, and the representation of cracking and postcracking anisotropic behavior dominates the modeling. The brittle cracking model is intended for applications in which the concrete behavior is dominated by tensile cracking and compressive failure is not important. The model includes consideration of the anisotropy induced by cracking. In compression, the model assumes elastic behavior. A simple brittle failure criterion is available to allow the removal of

elements from a mesh. The concrete damaged plasticity model is based on the assumption of scalar (isotropic) damage and is designed for applications in which the concrete is subjected to arbitrary loading conditions, including cyclic loading. The model takes into consideration the degradation of the elastic stiffness induced by plastic straining both in tension and compression. It also accounts for stiffness recovery effects under cyclic loading. The concrete damaged plasticity model is used for the concrete in the study.

CHAPTER 5

REINFORCED CONCRETE BEAMS UNDER ELEVATED TEMPERATURE DUE TO FIRE

In this chapter, the coupled thermal and structural analysis framework is used to simulate the response of reinforced normal strength concrete (NSC) and high strength concrete (HSC) beams under fire loading. Experimental results in the form of thermal measurements inside beam section and deflection with applied external temperature are used to compare with results from proposed computational model. The first section describes the experimental set up and test procedure and general model description, such as mesh size, elements, and thermal convection boundary conditions. The second section deals with parametric studies to examine the influence of material parameters, upper and lower bounds, on the thermal and mechanical solutions. The thermomechanical models for both NSC and HSC beams are presented in the last section.

5.1 Time thermomechanical modeling of reinforced concrete beams

This section first describes the nonlinear transient heat analysis performed with temperature dependent thermal material properties to generate the temporal and spatial temperature distributions for tested concrete beams. The temperature distributions are obtained from a 3D nonlinear transient heat analysis. Next, thermomechanical analysis is conducted with temperature dependent while the temperature is spatially prescribed at all times from the first heat analysis. For the transient heat analysis, 8-noded 3D diffusive heat transfer elements are adopted and the mesh is refined to match the locations of thermocouples and reinforcements used in the experiments. The thermal energy from the known remote heat sources is applied into the surface of the concrete beams through convection heat transfer. Therefore, a convection heat flux is applied as a boundary

condition on the three exposed surfaces of the beams. The applied convection heat flux is linearly related to the surface temperature differences through a convection coefficient (h) and described by:

$$q_c = h(\theta - \theta_\infty) \tag{5-1}$$

The range of h for air medium can vary from 2W/m^2°C to 25W/m^2°C. The remote temperature, θ_∞, is the ambient temperature. In this study, we follow the ISO-834 heating curve shown in Figure 5-1. The nodal temperatures obtained from the heat transfer analysis are used sequentially in the thermal stress analysis conducted while the beams are subject to constant total load of 88.8kN and 98.2kN for normal and high strength concrete beam as performed in the tests respectively. In thermal stress analysis, 8-noded 3D displacement brick elements are used with the same mesh refinement and time steps as the previous heat transfer analysis. The thermal and mechanical material properties for NSC and their temperature dependence are obtained from the literature, e.g. ACI report [1] and Harmathy [28, 29]. These include the effective specific heat (C_{eff}), conductivity (k), coefficient of thermal expansion (CTE), degraded elasticity modulus, and compressive stress-strain relationships. The material properties used in this study are shown in chapter 3.

Experimental results for reinforced NSC and HSC beams under fire are used to verify the analytical work. The dimension of the tested beams is 0.25×0.4m in cross section and 5m in span. Figure 5-2 shows the detail of the cross section with reinforcements and three thermocouples placed at the mid-span section to obtain temperature data during fire tests. The 28-day compressive strength of NSC and HSC is tested from cylindrical specimens and found to be 21.42 MPa and 57.85 MPa, respectively. The fire test set up for the concrete beams is shown in Figure 5-3. The detailed experimental works on NSC and HSC beams can be found in Lee [52], Shin [54], Shin et al. [55], and Kim [37]. Gas is used as fire source with flames through the internal

chamber for maximum internal temperature uniformity. The beam is loaded by four-point loading and three Linear Variable Differential Transducers (LVDTs) are used to measure the deflection with time. Insulation is used on the top portion of the beam, such that the bottom and two side surfaces are exposed to the thermal convection loading from the fire. Figure 5-1 describes the applied temperature curve, which represents a typical temperature load inside a building during fire condition developed by International Standard Organization (ISO) code 834. The tested concrete beams are heated using this curve until the maximum deflection of each beam is equal to L/24. During these tests, the temperatures within the mid cross-section are measured from the three thermocouples, and vertical deflections at the center and 1/4 of beam span are measured by three LVDTs.

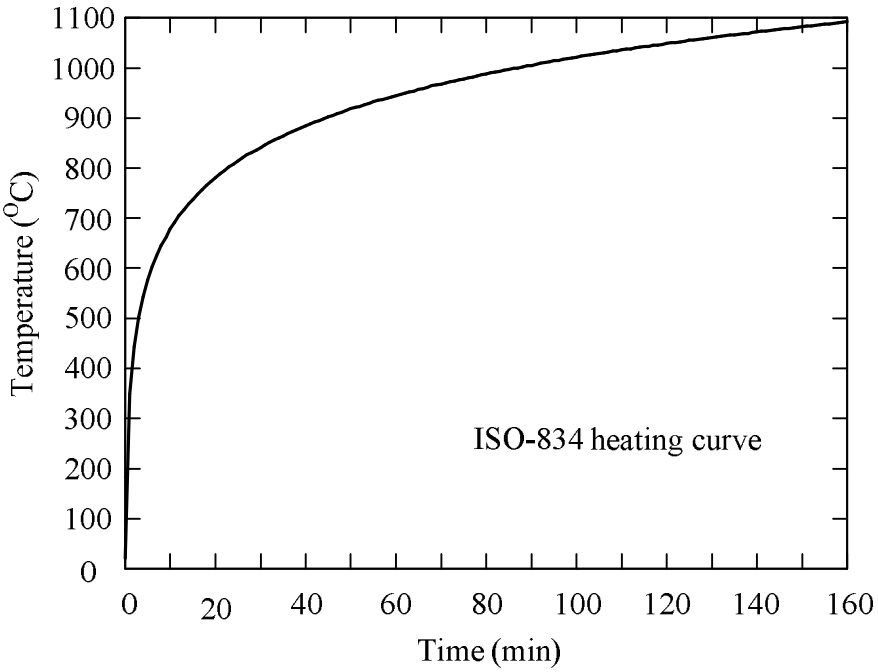


Figure 5-1. ISO-834 time dependent heating curve

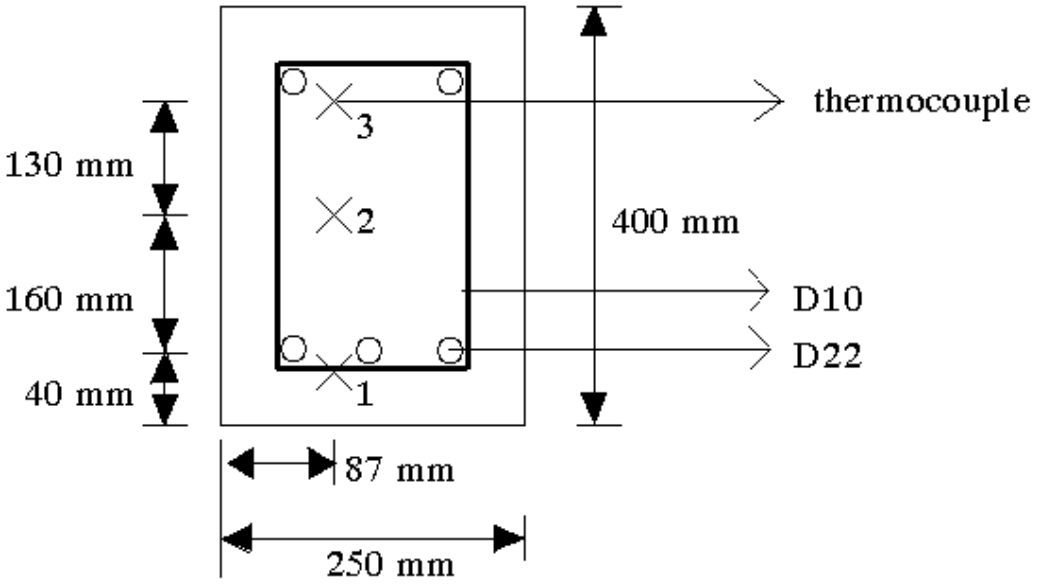


Figure 5-2. Section of concrete beam (tests conducted by Kim [37])

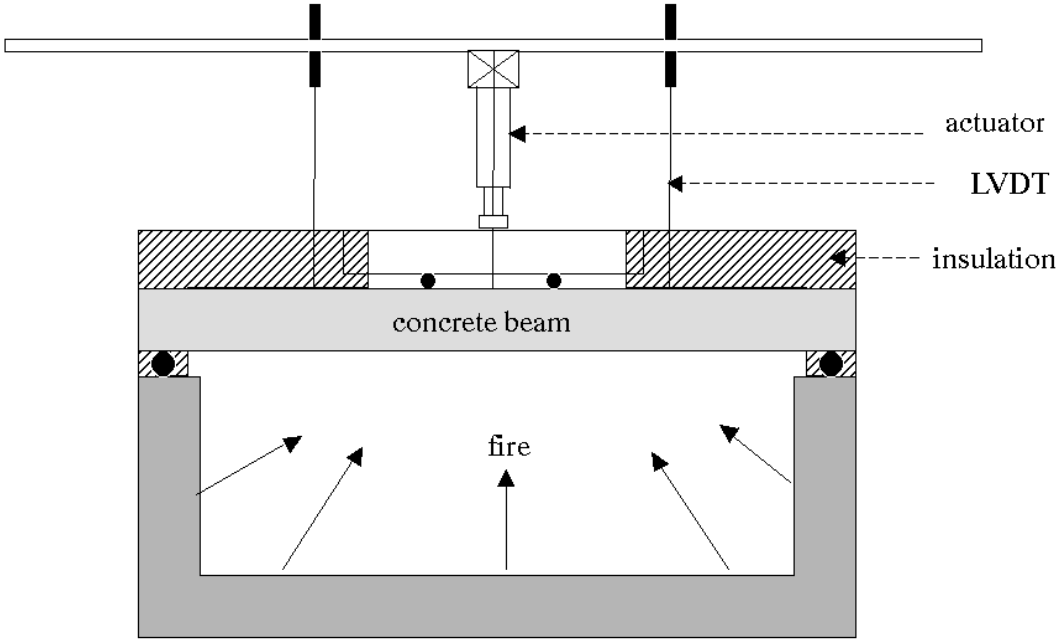


Figure 5-3. Set-up for fire test on concrete beam (tests conducted by Kim [37])

5.2 Parametric studies

The goal of this section is to identify the effect of thermal and mechanical material on the overall response of the concrete beams. The modeled thermal and mechanical material properties for concrete are assumed to be between the previously identified upper and lower bounds and selected based on the aggregate types. Toward that goal, the described thermomechanical problem for concrete beam is numerically analyzed with different range of material properties. The first parametric study is performed to examine the surface temperature solution using the extreme values of convection coefficient for air, namely $2W/m^2°C$ and $25W/m^2°C$, respectively. Figure 5-4 shows the surface temperature solution from the computational models as a function of time for the two extreme values of convection coefficients. It shows that convection has a great effect on the surface temperature and amount of energy absorbed by the beam. The second parametric study is used to examine the thermal response when using upper and lower bound values for the effective specific heat (C_{eff}). The convection coefficient and conductivity are fixed to $20W/m^2°C$ and $2.4W/m^2°C$, respectively. Temperatures at the center of cross-section of the beam are examined. Figure 5-5 shows the temperature results from transient heat analysis observed at the center of the beam. There is about 10% temperature difference when upper and lower bounds of effective specific heat capacity are applied, while a 20% higher temperature is shown when constant value of C_{eff} are taken at the room temperature and used throughout the simulation. An average temperature-dependent specific heat is generated using the two curves of upper and lower bound, since the temperature difference is small. This new averaged curve is used in subsequent analysis for NSC materials. The third parametric study is performed to study the effect of upper and lower bounds of conductivity on the temperature distribution. The convection coefficient is set to $20\ W/m^2°C$ and the previous averaged function of the effective specific heat is used. Both upper and lower bounds of conductivity for NSC

materials are applied in this parametric case as shown in Figure 5-6. After one-hour of heating, the temperature obtained from the analysis due to upper bound is about 61% higher than the lower bound temperature. Next, a parametric study is conducted to investigate the effect of different CTEs and degradation ratios of elastic modulus on the mechanical behavior of NSC beam depending upon choosing siliceous or carbonate based aggregate. Thermal Stress analysis is performed to examine the mechanical behavior of NSC beam. Figure 5-7 shows center deflection of NSC beam as a function of time when the beam is modeled with CTEs and degradation ratios of elastic modulus for carbonate and siliceous based aggregate. The mechanical difference in behavior of fire damaged NSC beams is small, due to similar thermomechanical properties between aggregate types. Figure 5-8 and 5-9 show results from parametric studies of different aggregate based material properties used in transient heat and thermal stress analysis for HSC beam, respectively. Transient heat and thermal stress analysis for siliceous aggregate based HSC beam show higher temperature distribution as well as larger center displacement than a carbonate aggregate based concrete. The thermomechanical material properties based on the siliceous aggregate are adopted for the modeling of both NSC and HSC materials. This is because HSC materials, with siliceous aggregates, when compared to that of the carbonate aggregates give a stable thermal solution. This is also seen when comparing both effective specific heats as a function of temperature.

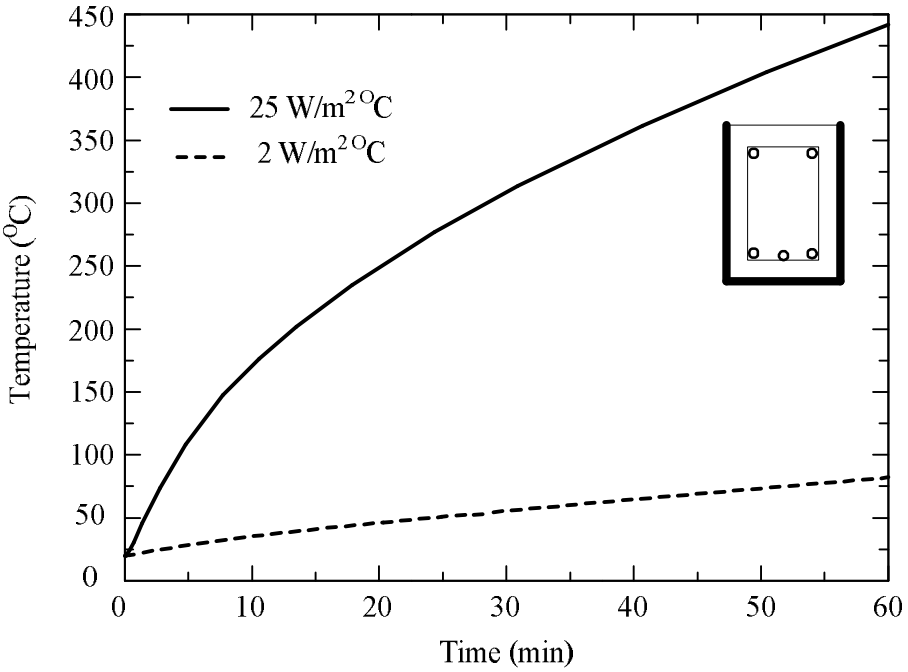


Figure 5-4. Surface temperature of NSC beam under different convection coefficients of 25W/m^2°C and 2W/m^2°C

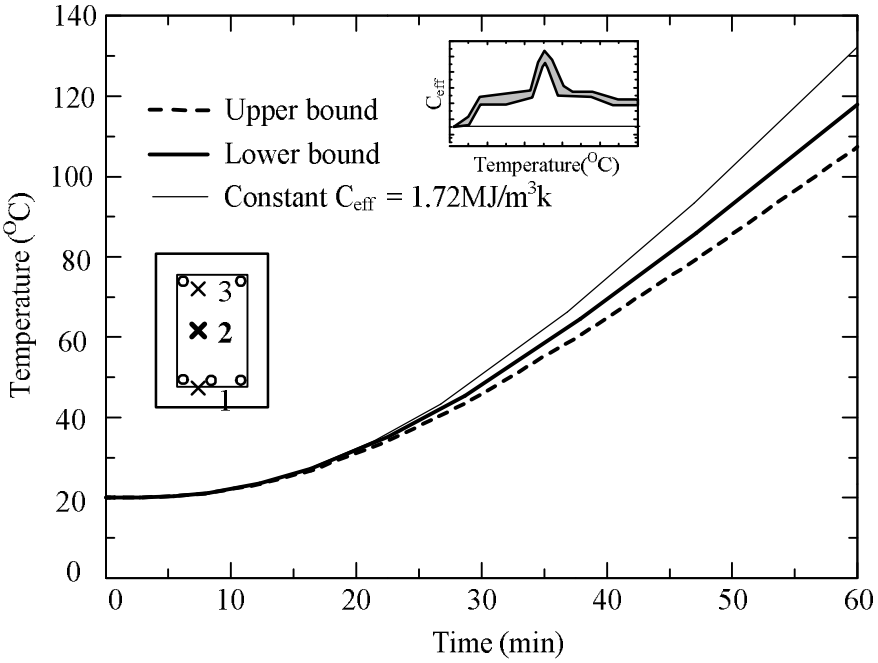


Figure 5-5. Temperature at the center of NSC beam section as a function of time from two thermal analyses using upper and lower bounds for the effective specific heat

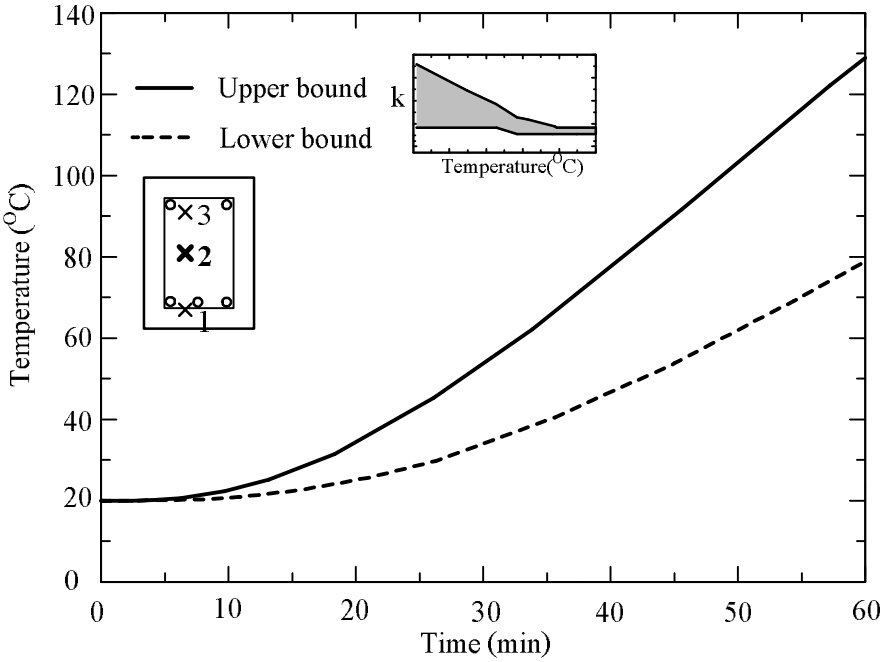


Figure 5-6. Temperature at the center of NSC beam section as a function of time for upper and lower bounds of conductivity

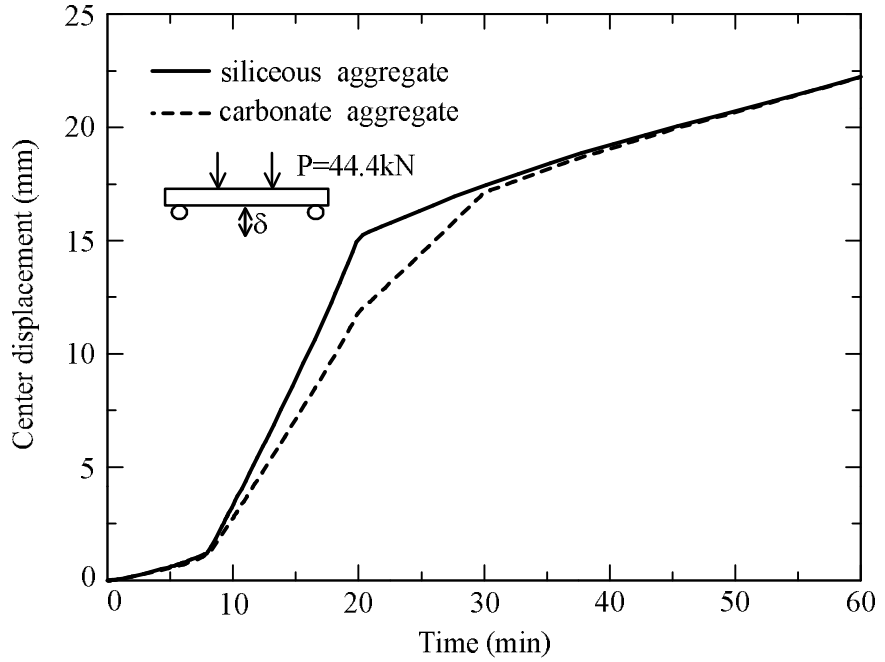


Figure 5-7. Deflections at the mid-span of NSC beam as a function of time due to different CTEs and elastic moduli depending on the aggregate type

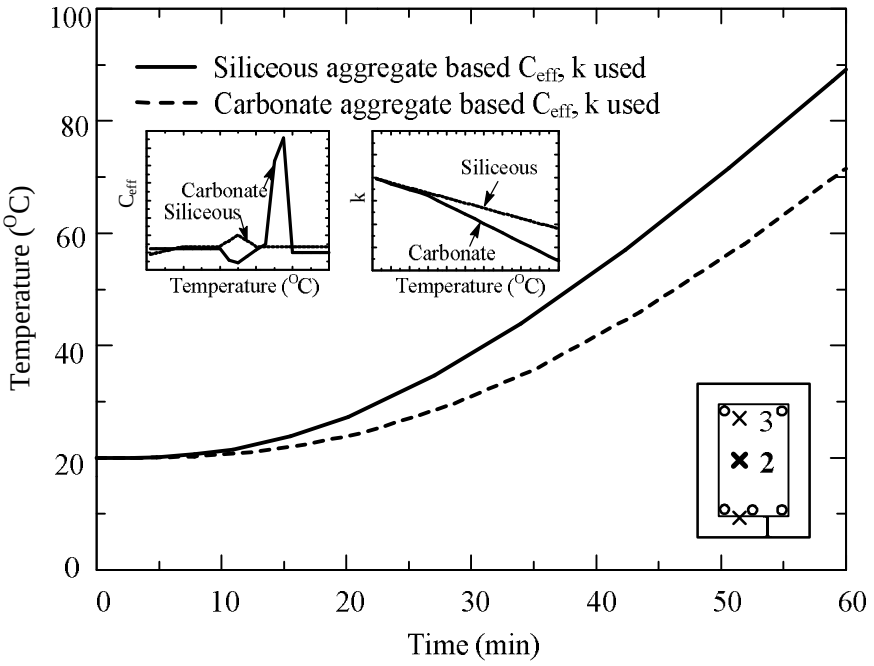


Figure 5-8. Temperature at the center of HSC beam section as a function of time for different aggregate based effective specific heat and conductivity

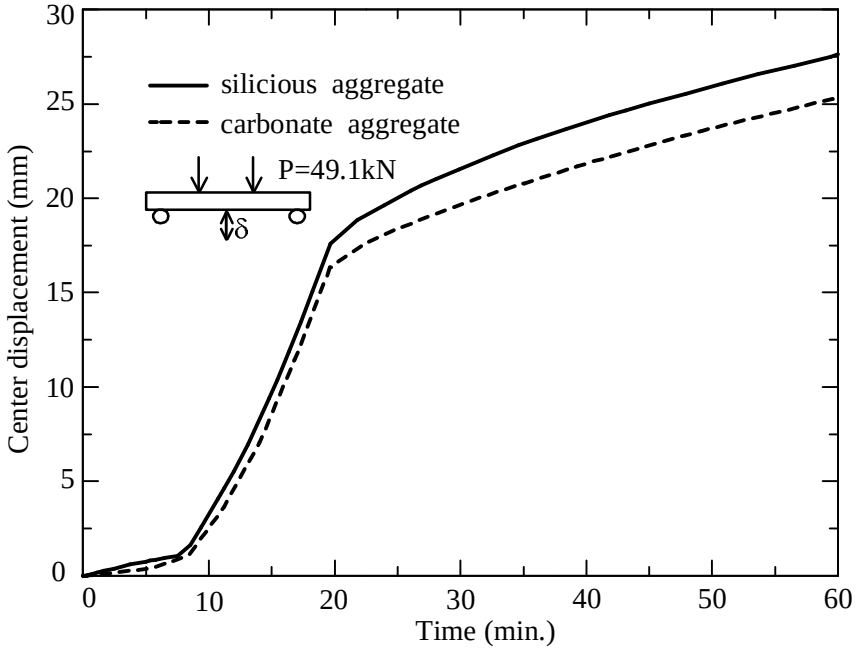


Figure 5-9. Deflections at the mid-span of HSC beam as a function of time due to different CTEs and elastic moduli depending on the aggregate type

5.3 Thermomechanical response and mechanical behavior of concrete beam under fire

This section presents the results of modeling the two NSC and HSC reinforced concrete beams under mechanical and fire loadings. Based on the parametric studies, a convection coefficient of 20W/m^2°C and averaged bound of effective specific heat and thermal conductivity are applied to the heat transfer analysis of the NSC beam. In the thermal stress analysis for both NSC and HSC materials, thermomechanical material properties, such as CTE and degradation ratio of elastic modulus are used based on the siliceous aggregate curves. Figure 5-10 shows the nonlinear transient heat temperature solutions through the section of NSC beam compared with experimental results measured at points 1, 2, and 3. Overall, the proposed model shows good agreement compared with the experimental results; however, the temperatures at point 2 and 3 obtained from the heat transfer analysis are higher than the experiments above the 75 minute time range. This can be explained by the moisture content of concrete beam. Large amount of moisture compared to a concrete structure members maybe present since the fire tests are conducted only three months after curing. The large amount of moisture inside the tested concrete beam is evaporated when the concrete is exposed to temperatures beyond 100°C. This delays the heat conduction. Figure 5-11 shows the deflections at the center and the 1/4 point of the beam-span obtained from the analytical model and experiments. As seen from the experimental results, the deflections increase rapidly after 90 minute of heating. The rapid increase of deflections can be attributed to the damage in concrete materials, especially softening due to cracking. Next, the tested HSC beams is analyzed and compared with the experimental results. Figure 5-12 shows the temperature distributions of the tested HSC beam as a function of time. The thermomechanical material properties for siliceous aggregates are used in the HSC model. As shown in Figure 5-12, the temperature distributions obtained from the analytical work are somewhat lower than the

experimental results. This difference can be explained from the coupling effect between heat conduction and mechanical material damage in this case. The damage due to spalling causes the decrease of cross section and increases the cracks on two the side surfaces. Figure 5-13 shows the mechanical response of the HSC beam measured at three different locations as a function of time. Pronounced deflection difference between the experiment and analysis exists after 90 minutes of heating, which can be explained by a heating accumulation due to cracking and spalling.

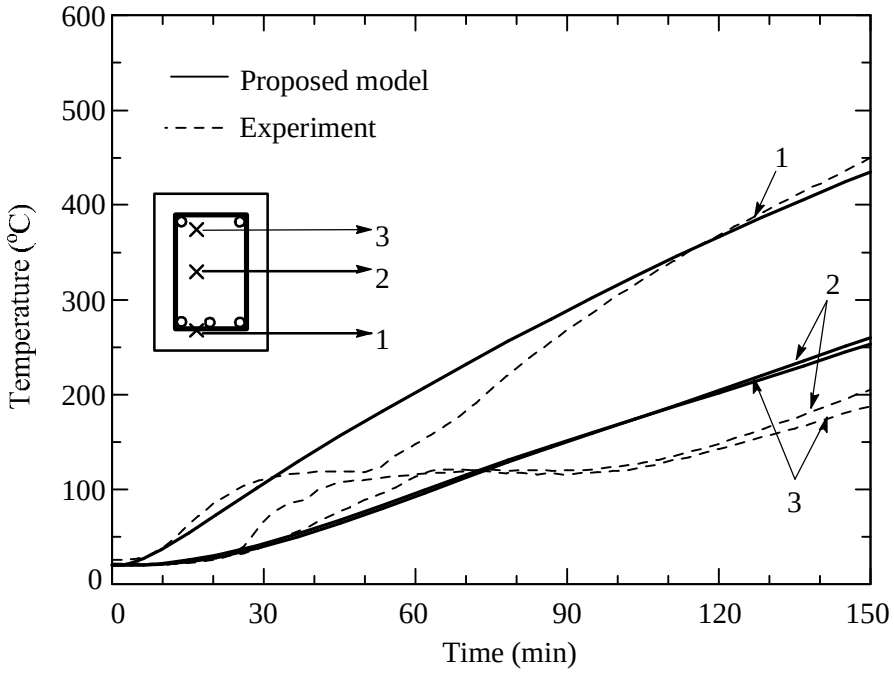


Figure 5-10. Comparison between analytical and experimental results of temperature distributions over a cross section of NSC beam

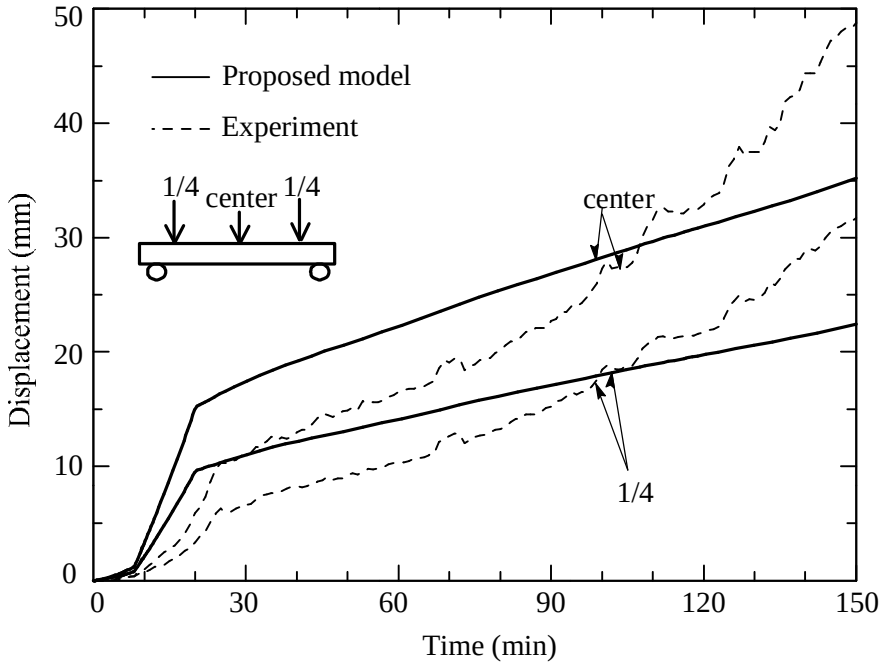


Figure 5-11. Analytical and experimental results of deflections of NSC beam under thermal and mechanical loading observed at center and 1/4 of beam span

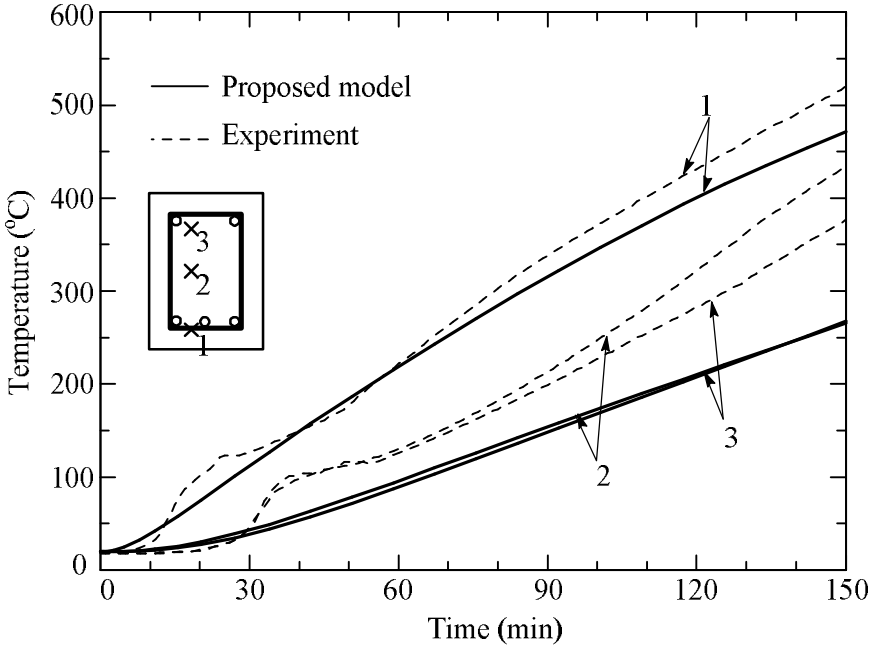


Figure 5-12. Comparison between analytical and experimental results of temperature distributions over a cross section of HSC beam

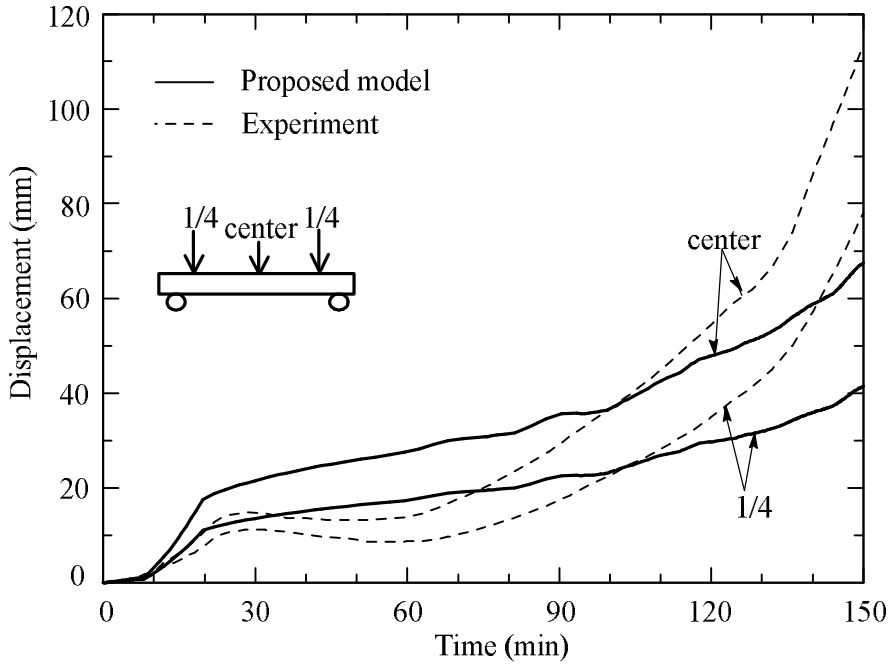


Figure 5-13. Analytical and experimental results of deflections of HSC beam under thermal and mechanical loading observed at center and 1/4 of beam span

CHAPTER 6

SIMULATION OF THE CARDINGTON FIRE TESTS

Case studies are performed and reported in this chapter in order to verify the proposed analysis framework. Two simulations are presented for two tests out of the four large-scale Cardington fire tests carried out by Building Research Establishment (BRE). The simulations are conducted for the third and the fourth fire tests compared with the available experimental data. The temperature and the deflection results are compared between predicted model and fire tests.

6.1 The Cardington fire tests

Full-scale fire tests on a realistic eight-story frame are carried out at the large building test facility at Cardington from September 1995 to June 1996. The overall objective of these tests is to increase our understanding for the complicated structural nonlinear and damage responses during fire. These tests highlight interactions role between different structural components as their local deformation determine the overall behavior of steel-concrete composite structures under fire. The Cardington test layout is shown in Figure 6-1. The cross-section of the building covers an area of 21m×45m and has an overall height of 33m. The beams are considered as simply supported with a composite action with the concrete slab. The structure is mechanically loaded using sandbags distributed over each floor to simulate a typical office dead load. Several fire tests are performed in Cardington using different structural system or compartments and fire sources. Among these tests, the third and fourth Cardington fire tests are investigated. This is because the fire scenarios used in the third and fourth tests are realistic and similar to actual office building fire events. Desks, papers, binders, and other office furniture are

used as fire sources in the tests instead the gas burners that are used in the first and second tests.

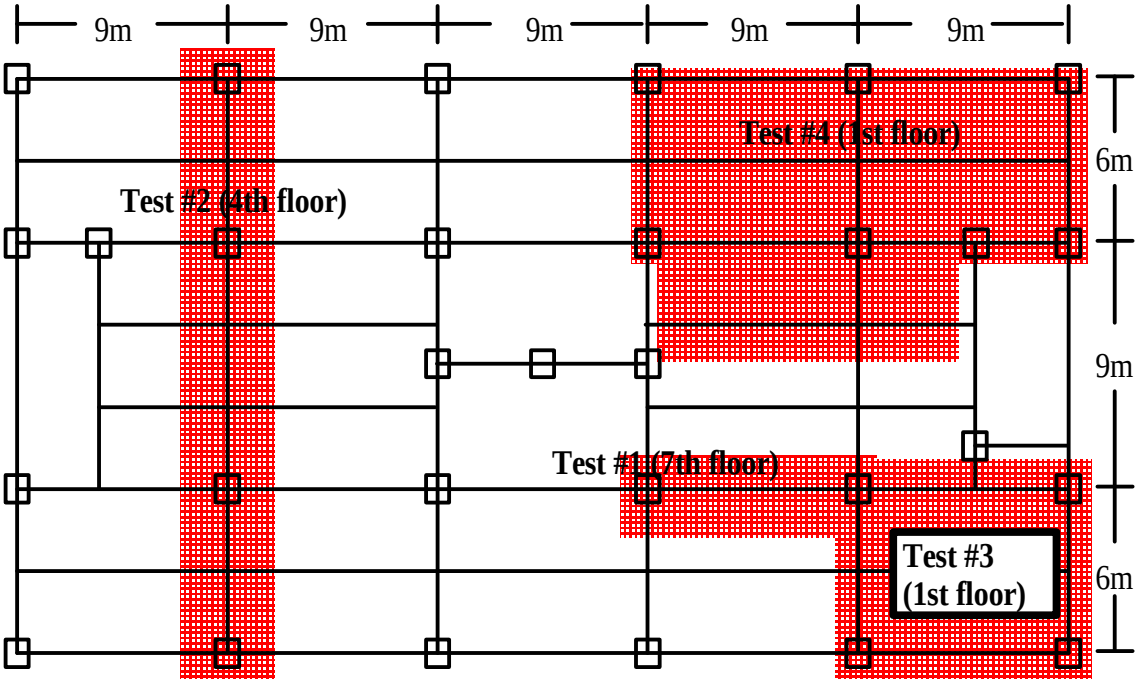


Figure 6-1. Layout of the British steel Cardington frame fire tests

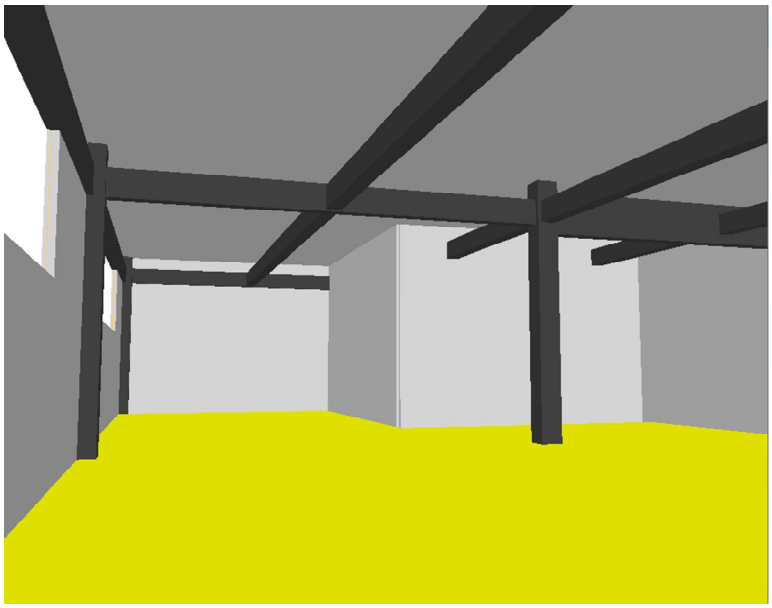
6.2 Proposed fire dynamics model for the fourth Cardington test

This test was conducted on the 1st floor using a concrete block compartment. The compartment floor area is about 135m^2 and a window area of about 25.6m^2 is used. The fire source consists of modern day furniture-like materials, such as computers, filing systems, wood, and plastic cribs. According to Newman et al. [49], the total material consumed in this severe fire is 46 kg/ m^2, made of 69% wood, 20% plastic and 11% paper. The columns and the beam-to-column connections are fire protected but all beams and beam-to-beam connections are exposed to fire in this fire test.

This section details the proposed fire model using FDS to simulate the Cardington fire during the fourth test. The overall objective of the FDS model is mainly to generate the temperature and heat results on the structural elements within the fire compartment.

Figure 6-2 illustrates the proposed refined FDS model of the fourth Cardington fire compartment along with the structural elements. The FDS is conducted mainly in order to obtain the surface temperatures on the structural elements. The yellow "floor" area is the distributed fire source that can be specified directly as a heat release rate per unit area (kW/m^2). The solid darker lines indicate the enclosed fire area while the other solid lines indicate beam elements. The solid circles indicate locations where the vertical displacements are measured. Newman et al. [49] details the different materials (wood, plastic, paper) used in the fourth test. This information is used in this study to estimate the FDS fire load curve as illustrated in Figure 6-3. The assumptions for this fire source is a growth rate of α=0.08 based on Karlsson and Quintiere [35]. The average heat density of consumed materials is taken as 33kJ/g assuming an upper limit of energy production based on the tables from Bwalya et al. [69] which provided itemized tables for heat combustion values of different materials with lower-medium-high range of heat generated in MJ/kg. This information is used in this study to create an average sum of the fire content consumed in the Cardington test. The fire curve used in the FDS simulation for the fourth Cardington fire test is shown in Figure 6-3. The maximum heat release rate is 1003 kW/m^2. Growth and decay phases are assumed where both are taken 2 minutes, while the steady-state phase lasts for 24 minutes. The FDS model is generated for the fire compartment and it did not cover the entire building. Other larger FDS models including top and bottom floors have been examined but did not significantly alter the surface heat and temperatures within the fire compartment. The steel columns and beams are added in the FDS model in order to capture the temperature profiles on their surfaces by post-processing the interior surface temperatures. Figure 6-4 illustrates the FDS model and typical surface temperature results in the form of a contour on a slice plane (the blue plane in the left figure) and heat release rate per unit volume HRRPUV (right).

The measured temperatures on beams inside the office area are compared to those generated from the simulation. A range of temperatures measured during the test on different locations is also added as upper and lower values reported by Usmani et al. [70] in the Figure 6-5 since the exact location of the reported experimental measurements are not known to us. The solid dashed line is a curve generated directly from the FDS discrete temperature results. The solid line is the results from the temporal-spatial polynomial fourth-order approximation of the temperature for the entire range of fire simulation. It is plotted to illustrate the quality of the polynomials used to represent the surface temperature in the nonlinear 3D thermal stress analysis. Figure 6-6 shows similar results to Figure 6-5, however, the temperature is a function of the distance from the window along the center beam instead of time scale. Note that two temperature measurements are available for this time instance.



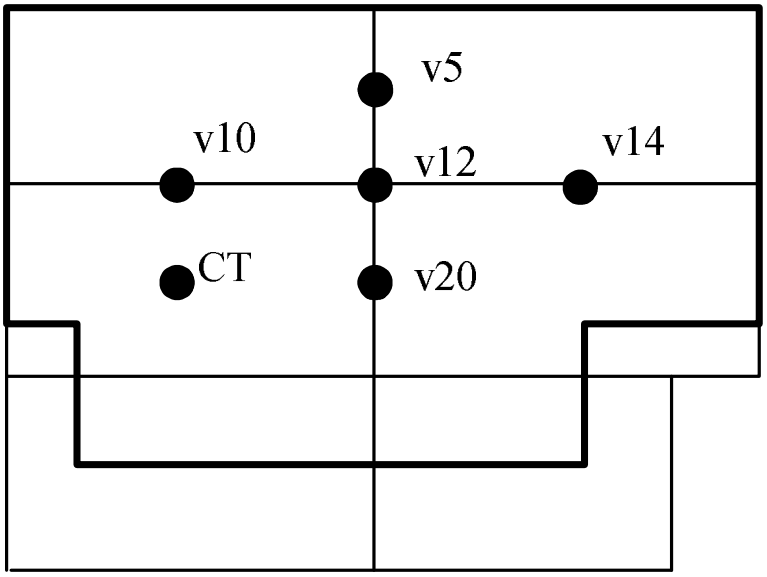


Figure 6-2. The Cardington office fire demonstration test modeled by FDS

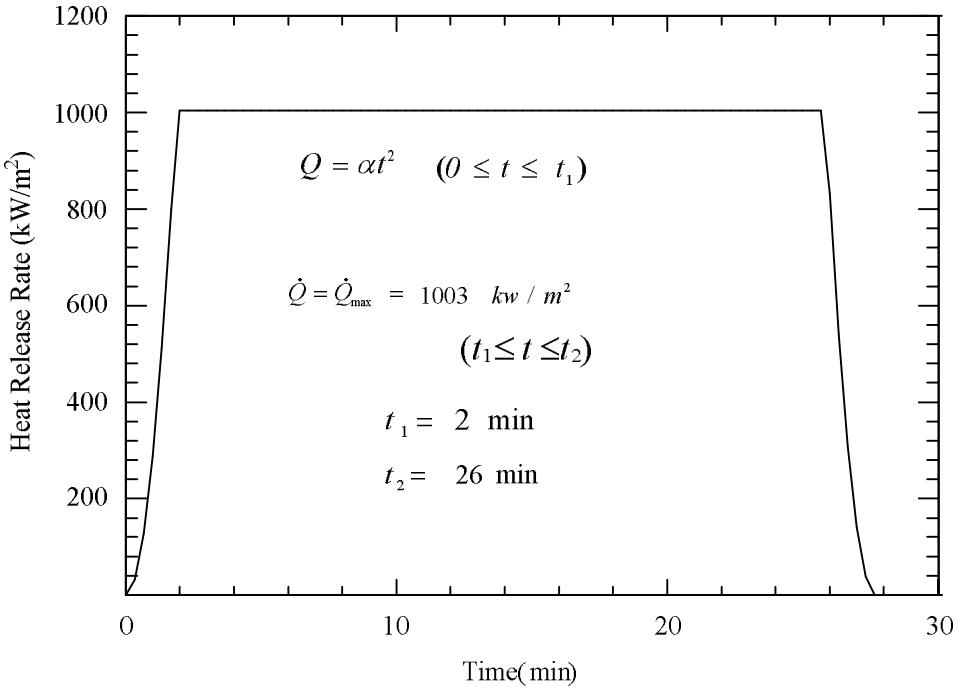


Figure 6-3. A fire load curve used to simulate the source in the Cardington office fire test

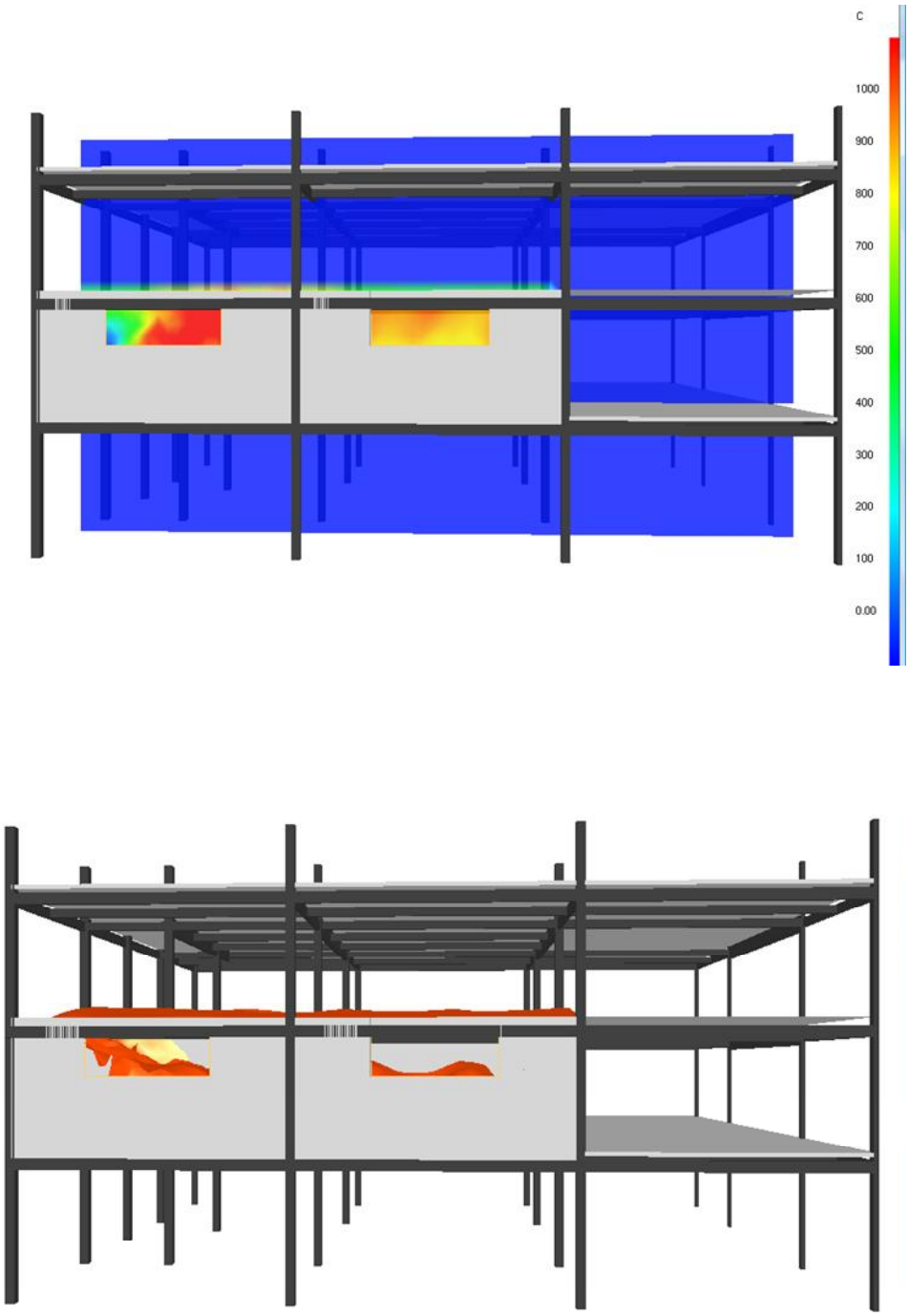


Figure 6-4. The temperature slice picture and HRRPUV during the FDS

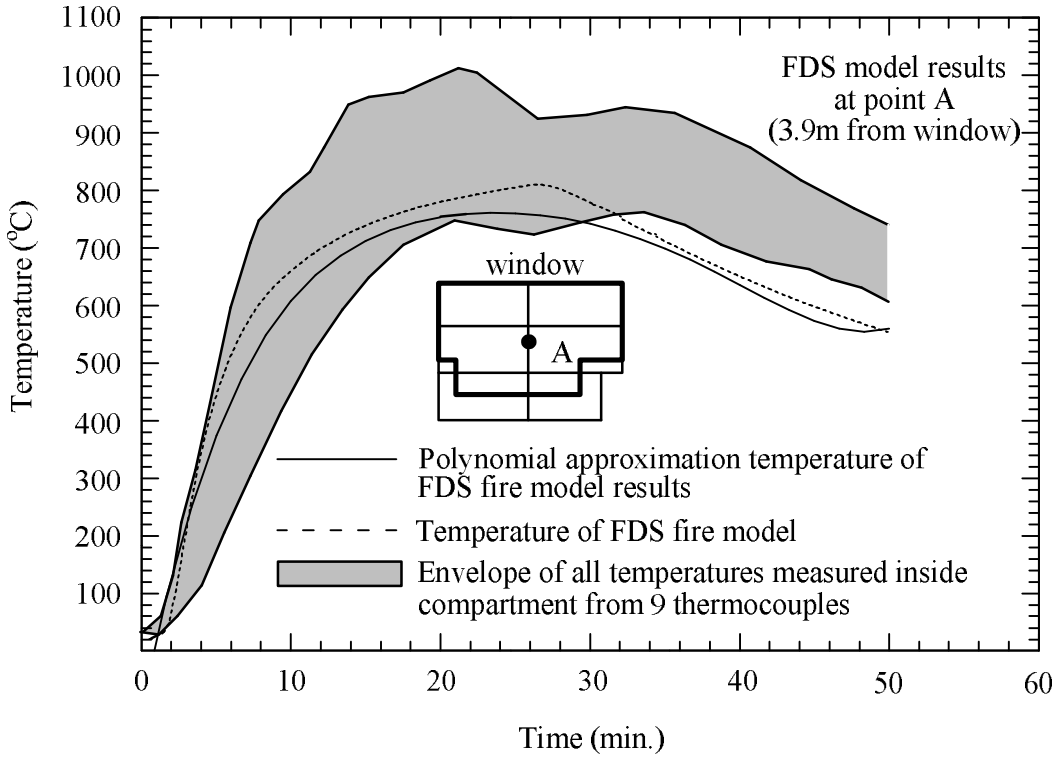


Figure 6-5. Polynomial approximation and FDS temperature time-values compared with experimental values on different beams in the office fire

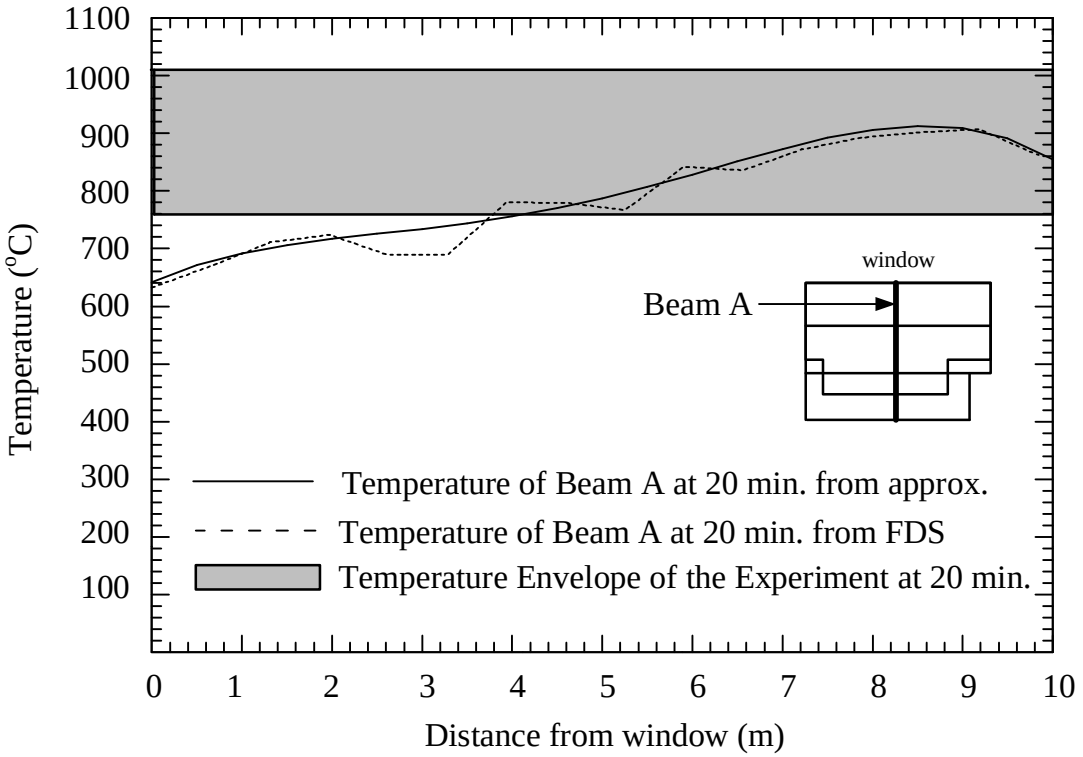


Figure 6-6. Comparing the temperature from FDS and approximation along a distance away from the window area

6.3 Transient heat model for the slab in the fourth Cardington test

A separate heat transfer analysis is proposed to generate the temperature profiles through the thickness for the concrete slab. The temperature results from the heat transfer analysis are needed to be applied to the concrete slab part of the nonlinear 3D thermal stress analysis. The heat transfer analysis for the slab in the fourth Cardington test is modeled using four-node quadrilateral shell elements (DS4). Figure 6-7 shows the FE model for the concrete slab. The red area is used for the fourth Cardington fire test. The number of element and nodes used in the model are 6372 and 7400, respectively. Only one floor considered as the ceiling floor is modeled for the analysis. The heat flux results from the FDS simulation are applied to the slab model on its bottom surface. A room temperature is applied to the top surface. Through the thickness temperature points are added (7 temperature points are used in current model) to generate the heat and temperature distribution through each shell element. The main reason for using this type of element in the transient heat, unlike 3D elements, is to have a similar shell element to the one used the subsequent structural analysis with same temperature points. Figure 6-9 through Figure 6-11 show predicted temperature results from our heat transfer analysis comparing with the temperature results of Lamont et al. [39] for inside concrete slab. Figure 6-9 shows the positions where the temperatures are computed through the concrete slab thickness. Figure 6-9 through 6-11 show the predicted values because measured temperature data for the concrete slab are not known. On the other hand, the inside concrete temperatures compared with experiments for the third Cardington test are shown in Figure 6-19 through Figure 6-21.

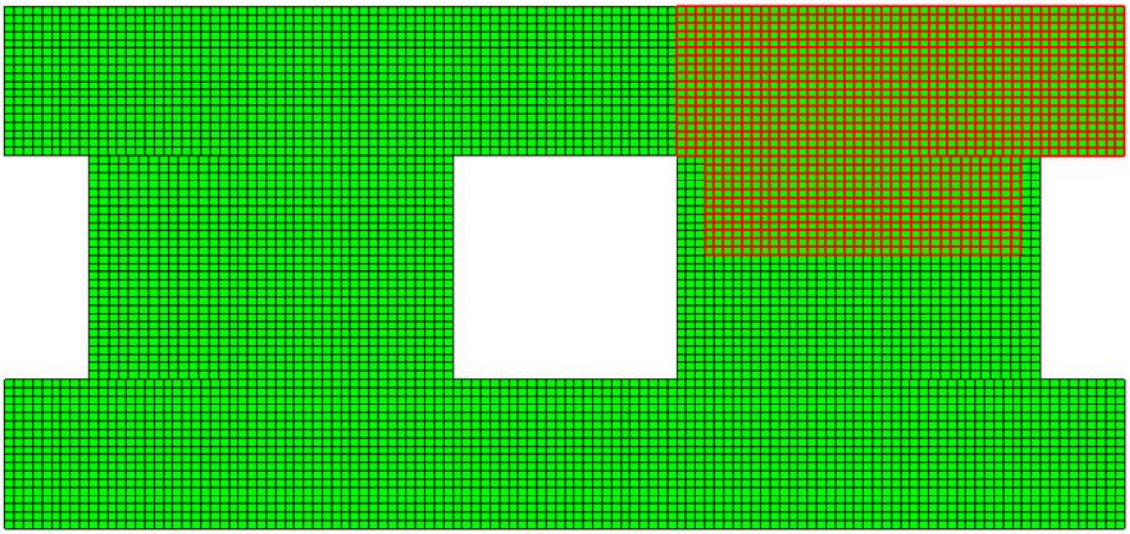


Figure 6-7. Heat transfer analysis FE model for the fourth Cardington test.

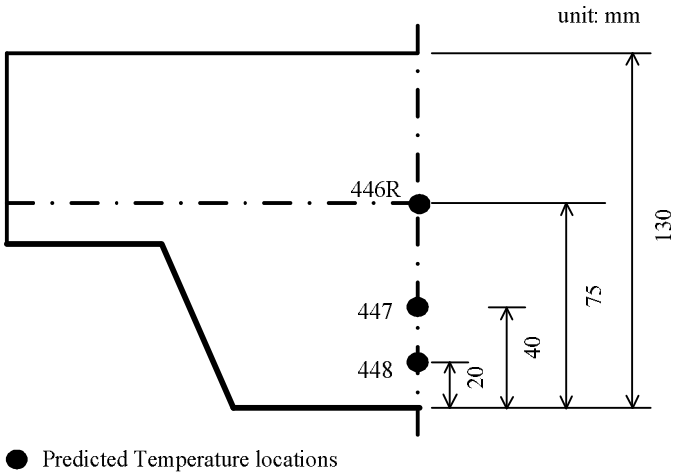


Figure 6-8. Cross-section of the concrete slab illustrating the locations (filled and numbered circles) where the temperature are reported.

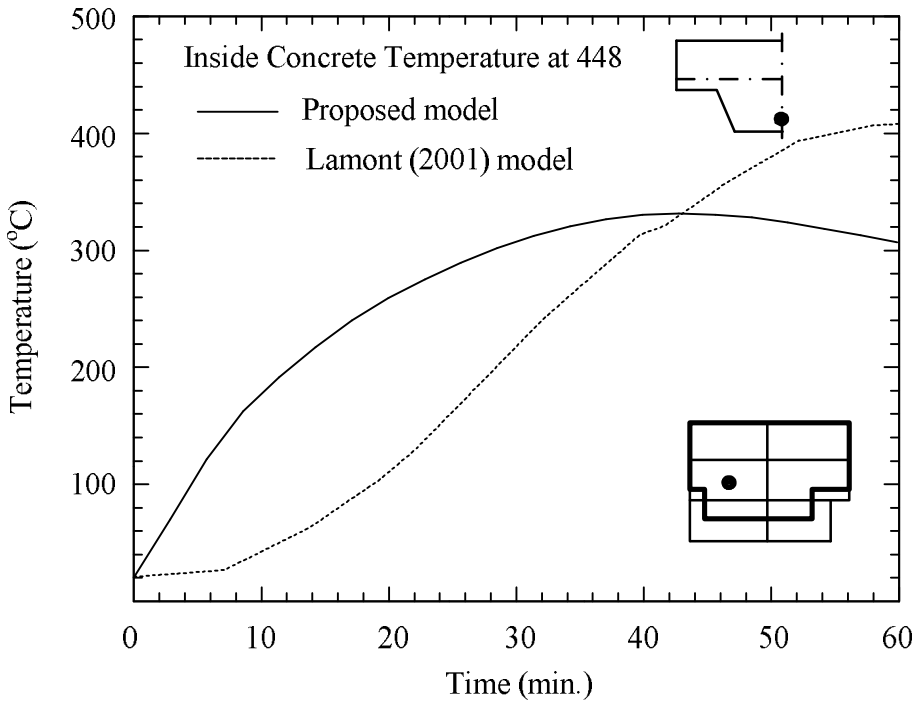


Figure 6-9. Temperature history results at a point (448) in the concrete slab from heat transfer analysis of the proposed model compared with Lamont et al. [39].

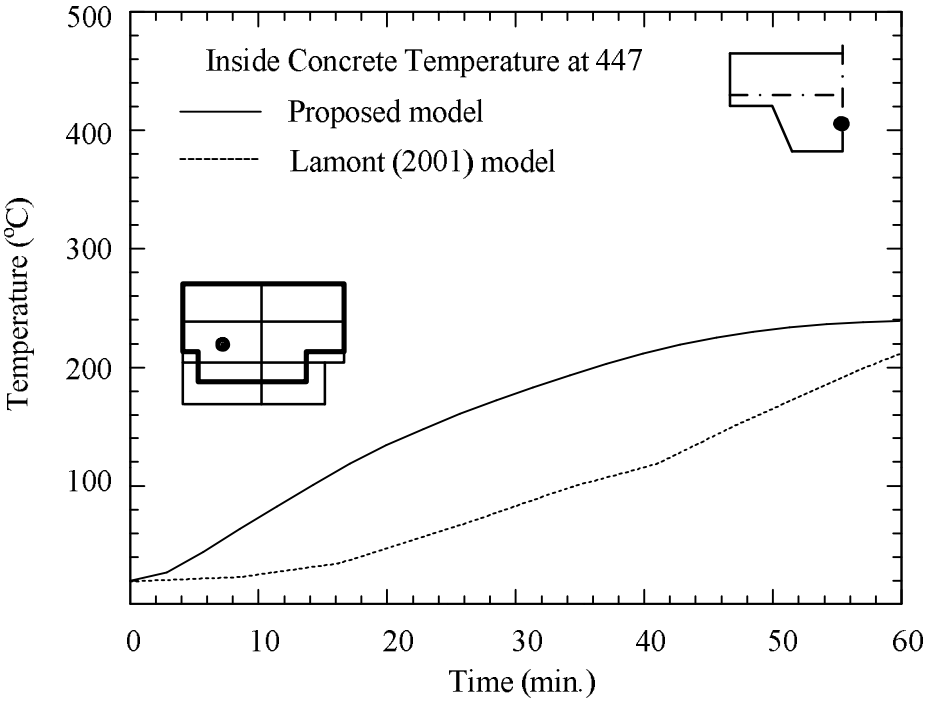


Figure 6-10. Temperature history results at a point (447) in the concrete slab from heat transfer analysis of the proposed model compared with Lamont et al. [39].

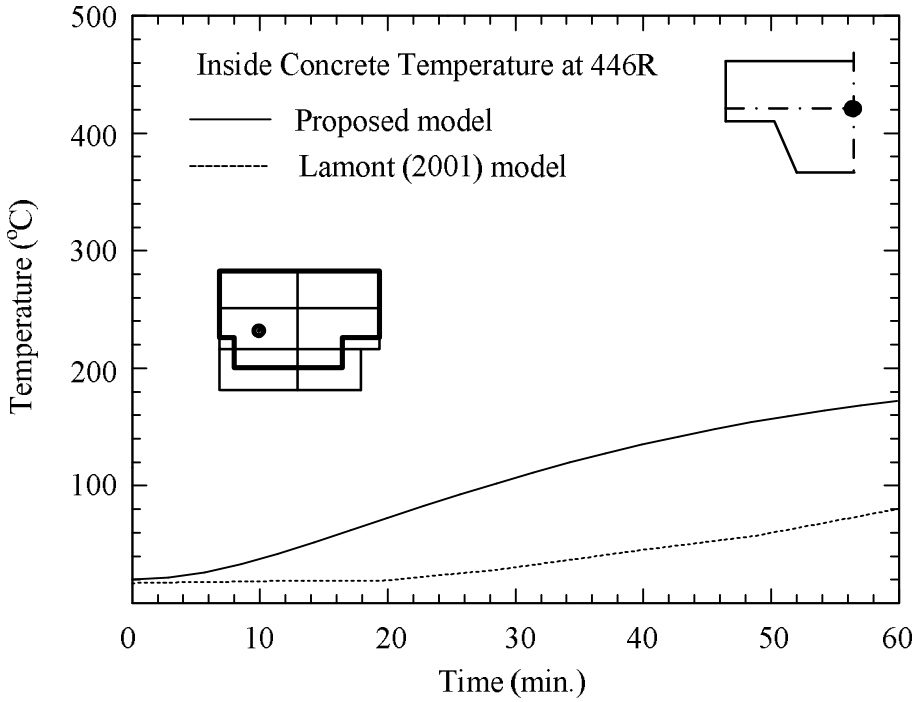


Figure 6-11. Temperature history results at a point (446R) in the concrete slab from heat transfer analysis of the proposed model compared with Lamont et al. [39].

6.4 Concurrent nonlinear thermal structural analysis of the fourth Cardington test

This section presents the nonlinear 3D thermal stress analysis results for the fourth Cardington fire test. The steel beams, columns, and concrete slab are the structural elements considered in the current model. The ABAQUS finite element (FE) model includes a 3D model with beams and columns using the two-node linear (B31) beam element and concrete slab by four-node shell element (S4R). Figure 6-12 and Figure 6-13 show the whole stress analysis model with a 3D view point and top view used for the fourth Cardington fire test. The half of the concrete slab is considered and the red rectangular represent the area of the fourth Cardington test in Figure 6-13. Number of elements and nodes used in the stress analysis model are 18441 and 42154, respectively. Both nonlinear geometry and incremental plasticity are included along with temperature dependent stiffness and hardening behavior. The material stress-strain temperature-dependent responses used in the current study for steel are reported by the Eurocode 3

[20], as shown in Figure 3-10. In addition, the concrete material stress-strain responses with different temperature employed in this study are given in Figure 4-5 and Figure 4-7. This analysis considers not only concrete compression but also concrete tension property. Figure 6-14 to Figure 6-18 compare the FE predictions to the measured displacements which is reported by Usmani et al. [70]. The results from a model by Usmani et al. [70] are also shown in the Figures. An explicit approach was applied to their model. The predicted deflections from proposed model capture the overall trends of the test results during both the growth and cooling stages of the fire. The results of Usmani et al. [70] are also reviewed. However, the latter only considered the growth stage in their analysis. Furthermore, they have not performed fire simulations prior to their transient heat analysis. Instead they applied a uniform surface slab temperature using the reported experimental data. In this study, we used the spatially predicted surface temperatures from the FDS model.

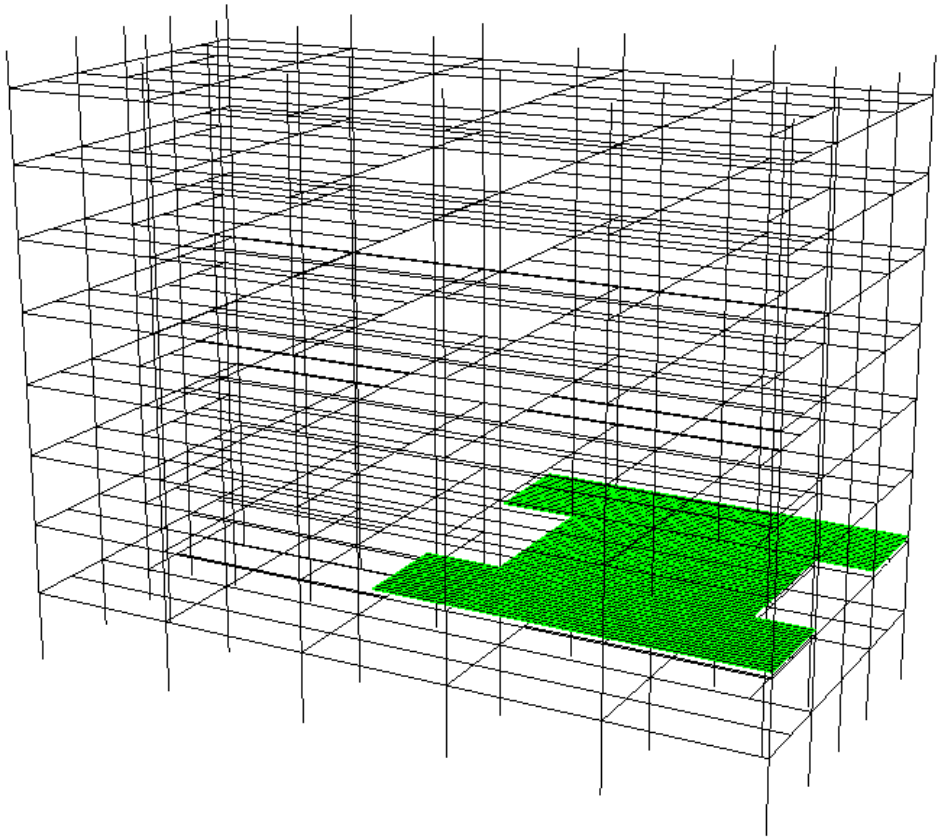


Figure 6-12. Stress analysis FE model for the fourth Cardington fire test (3D view)

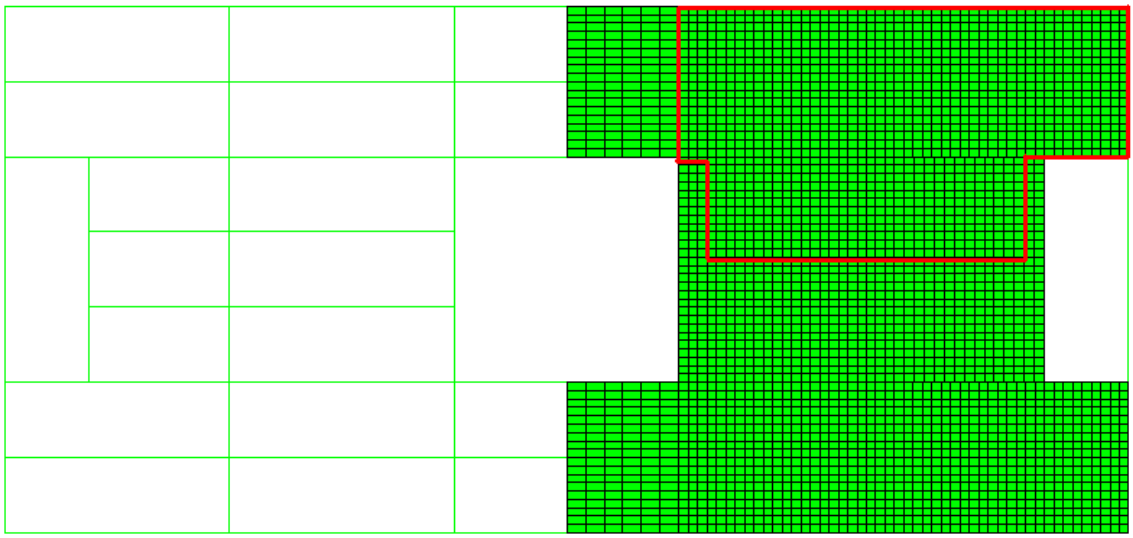


Figure 6-13. Stress analysis FE model for the fourth Cardington fire test (top view)

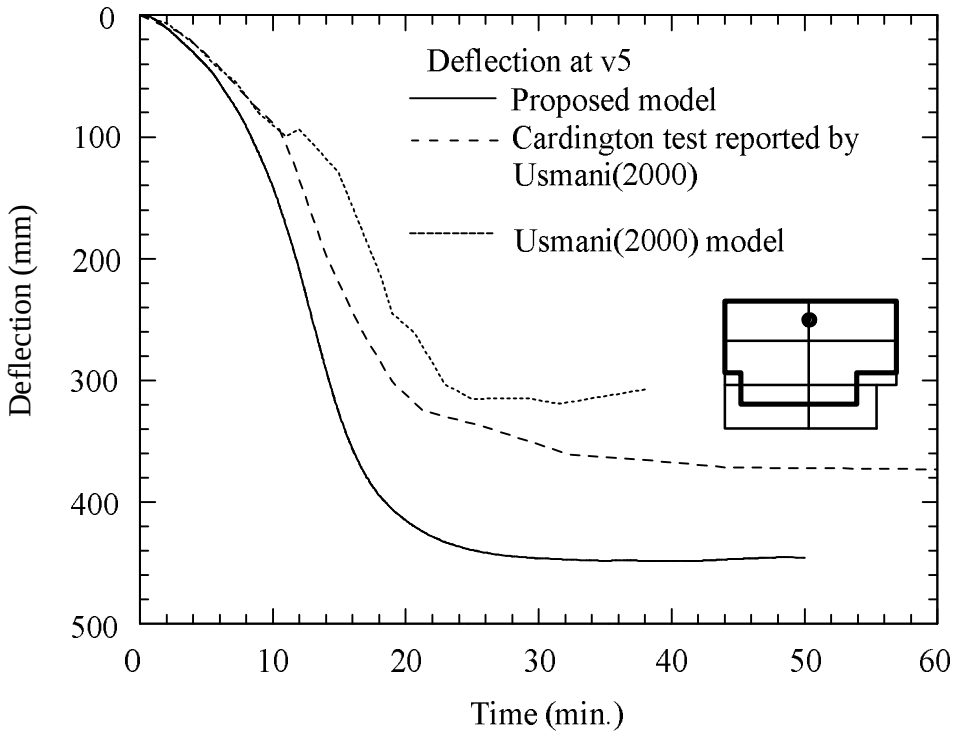


Figure 6-14. Predicted beam deflection history (at the circled point, v5) compared with the reported experimental data and the model proposed by Usmani et al. [70].

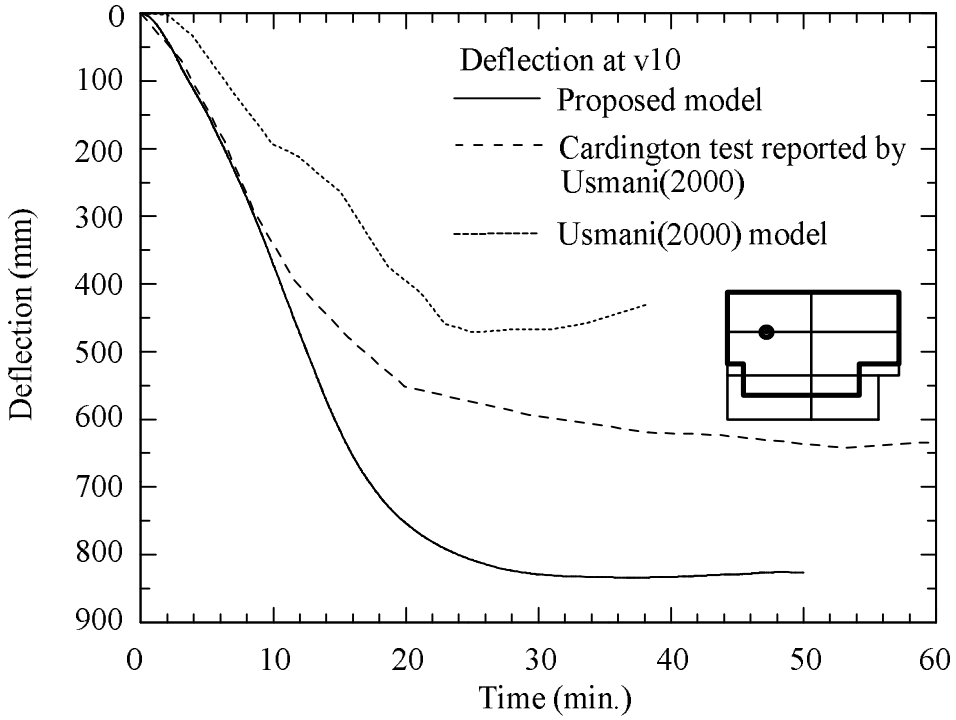


Figure 6-15. Predicted beam deflection history (at the circled point, v10) compared with the reported experimental data and the model proposed by Usmani et al. [70].

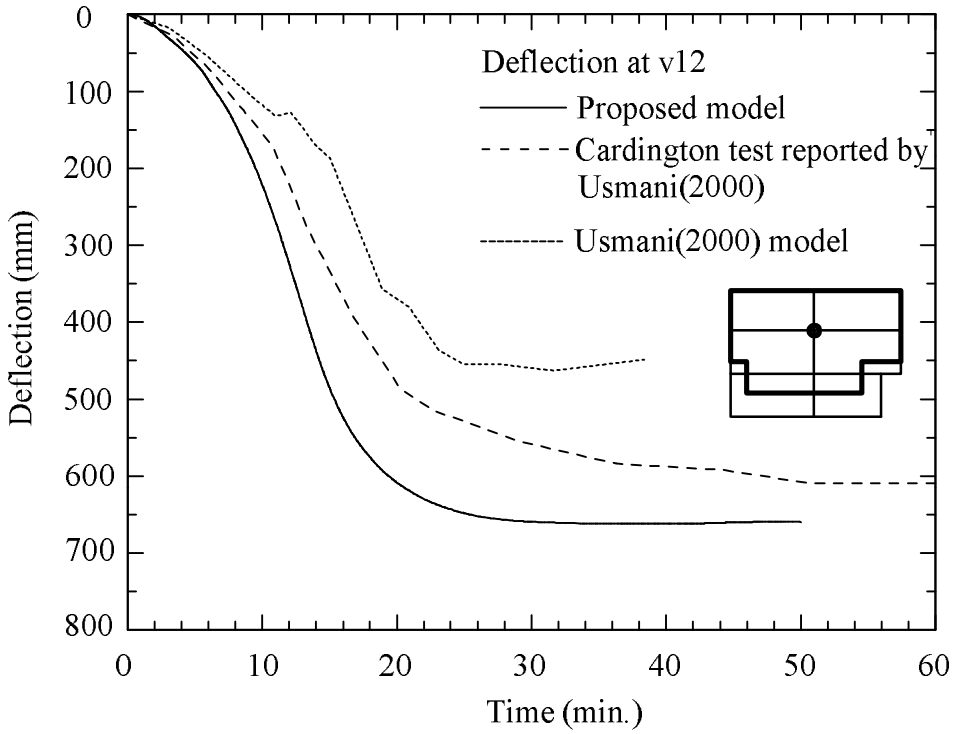


Figure 6-16. Predicted beam deflection history (at the circled point, v12) compared with the reported experimental data and the model proposed by Usmani et al. [70].

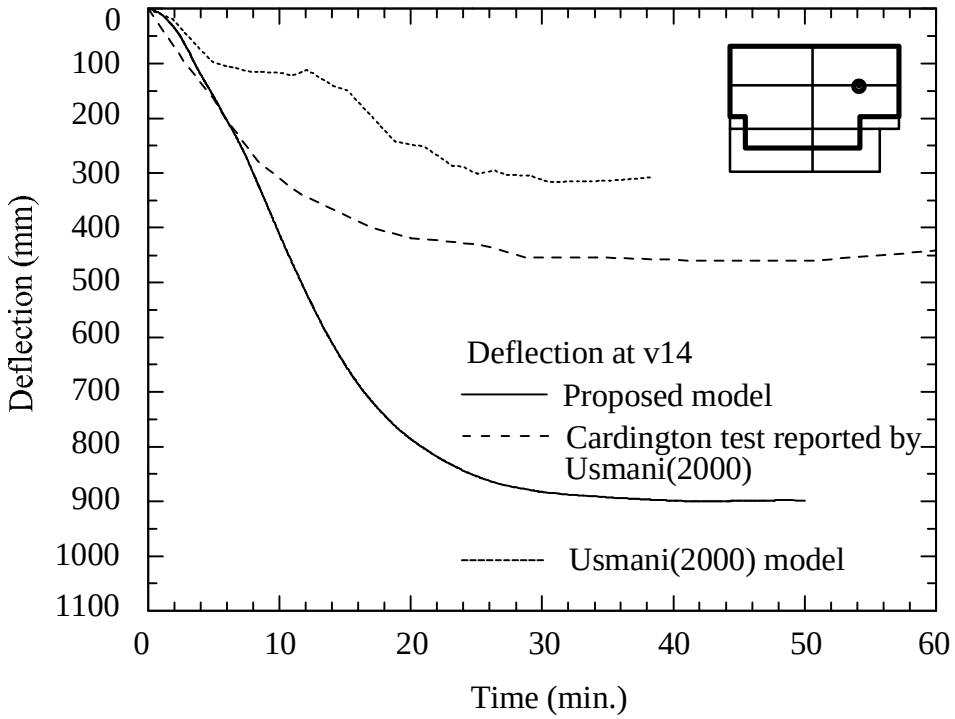


Figure 6-17. Predicted beam deflection history (at the circled point, v14) compared with the reported experimental data and the model proposed by Usmani et al. [70].

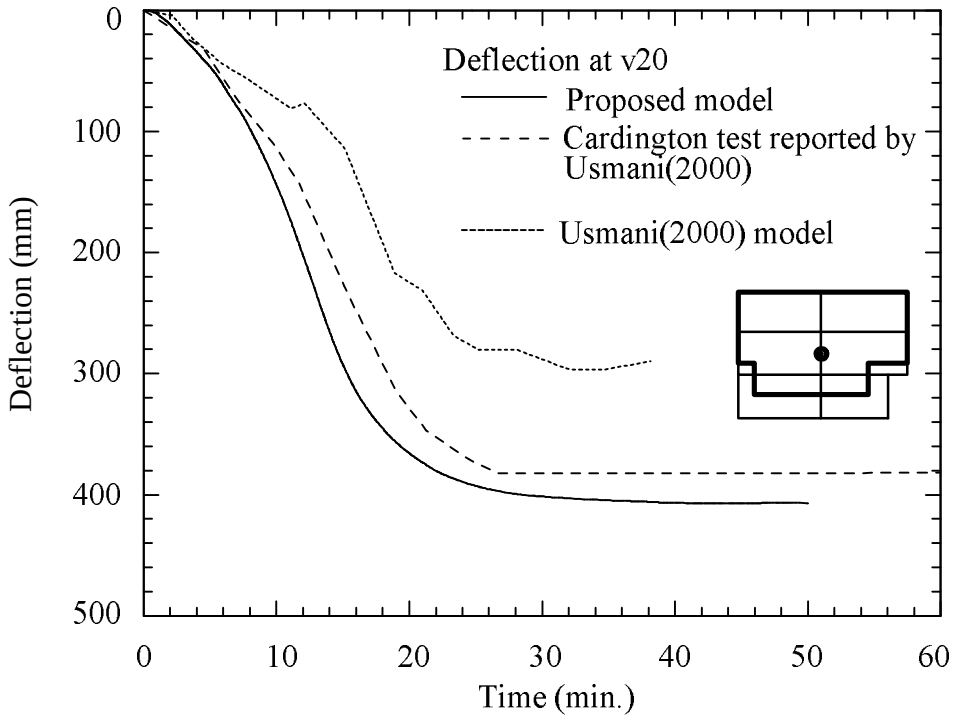


Figure 6-18. Predicted beam deflection history (at the circled point, v20) compared with the reported experimental data and the model proposed by Usmani et al. [70].

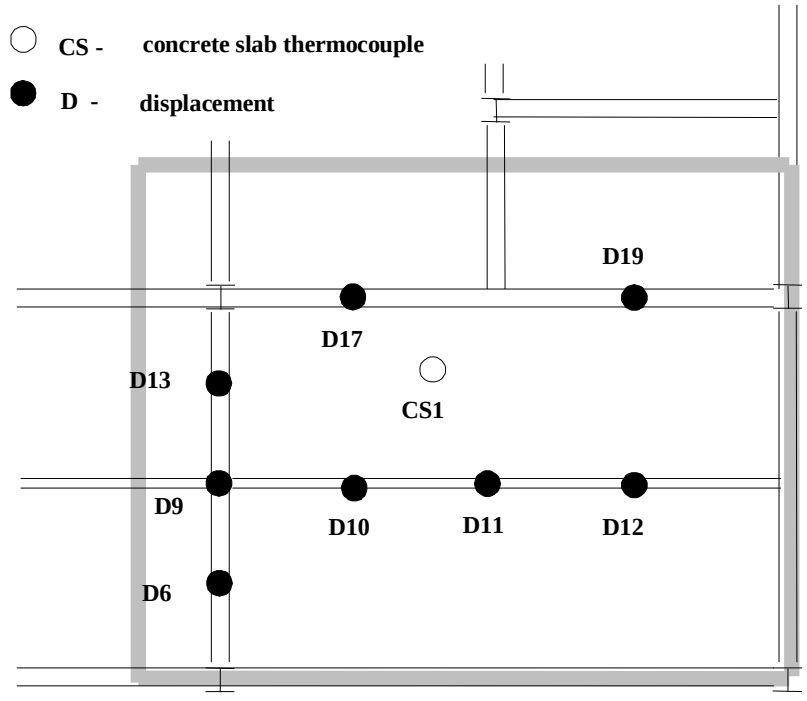
6.5 Proposed fire dynamics model for the third Cardington test

The third Cardington fire test is also performed on the first floor of the building used to study the behavior of a complete floor system. The compartment area is approximately 80m^2 and located at the South East corner of the building as shown in Figure 6-1. A realistic fire is created by consuming wood-based material with an estimated fire loading of 45 kW/m^2 as reported by Newman et al. [49]. Ventilation is provided by a single 6.6m×1.8m high opening window. All columns, beam-to-column connections, and perimeter beams were fire protected. The secondary beams are equally spaced and have 9m spans connected to the columns or to the primary beams. The heated primary beam has a length of 6m. All columns are fire protected along their full height. The composite profiled deck slab has a span of 3m between the secondary beams (Figure 6-1).

The proposed fire modeling using FDS is applied to simulate the third Cardington fire test. The FDS model is mainly used to generate the temperature and heat histories on the surface of the structural elements within the fire compartment. Figure 6-19(a) shows the FDS model for the third Cardington fire test along with structural elements. Similar to the fourth Cardington fire test, the "yellow floor" area is used to distribute the fire source. The grey solid objects represent the concrete slabs and the walls. The black solid elements represent the steel columns and beams. Figure 6-19(b) shows the locations where the FE predictions for the displacements and temperatures are compared with the reported experimental data. The temperatures through the thickness of the slab from the transient heat transfer analysis are compared at the location CS1. The other black filled circles are used for displacement comparisons. The grey rectangular indicates the third Cardington fire test compartment boundary. The applied heat release rate (HRR) is assumed to have a growth rate factor α of 0.0007 based on Karlsson and Quintiere [16] and a maximum heat release rate per unit area of 162 kW/m^2. These values are used to define the fire source by comparing predicted temperature response in the compartment to averaged measured temperature. The fire load curve used for the third Cardington fire test simulation is shown in Figure 6-20. The duration time for growth phase, steady-state phase and decay phase are 8 min, 73 min and 8 min, respectively. The interior surface heat flux profile from the FDS model is applied to nonlinear 3D transient analysis for the concrete slab. The temperature from FDS model results are used to generate continuous ST functions used in the thermomechanical structural analyses of steel beams and columns.

Figure 6-21 compares the predicted FDS temperature results with the measured beam temperatures inside the third Cardington test area. The range for the measured beam temperatures is reported by Usmani et al. [25]. The FDS model results show temperatures for three different locations: A, B, and C. The FDS model shows good prediction for low to moderately high temperatures relative to the experimental values.

(a) Fire dynamics simulator model



(b) Locations where the FE predicted

Figure 6-19. The Cardington corner fire test modeled by FDS

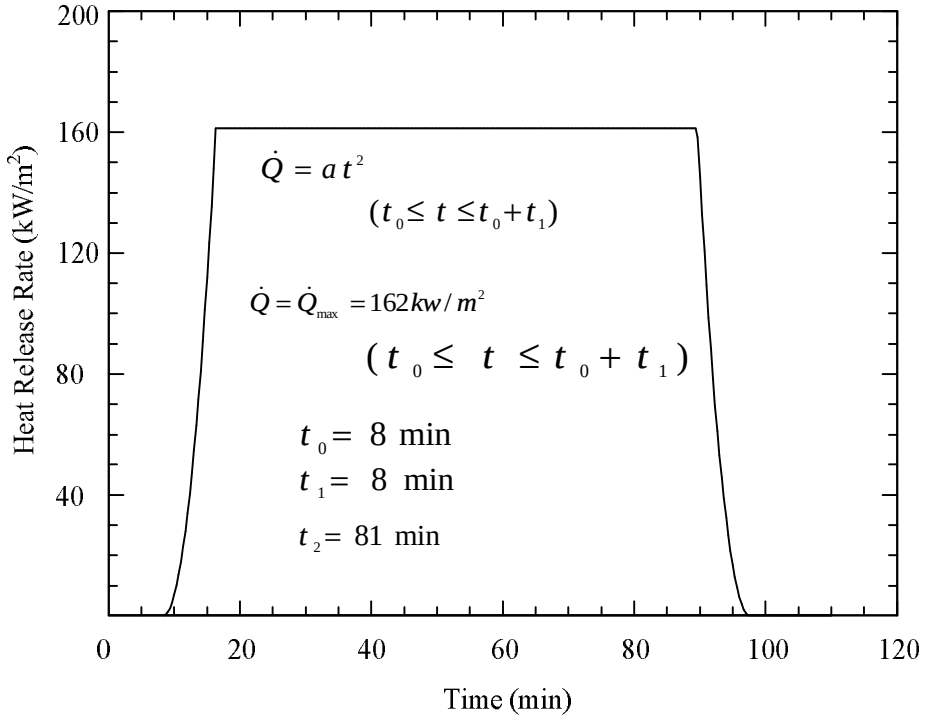


Figure 6-20. Fire load curve used to simulate the FDS source in the third Cardington fire test

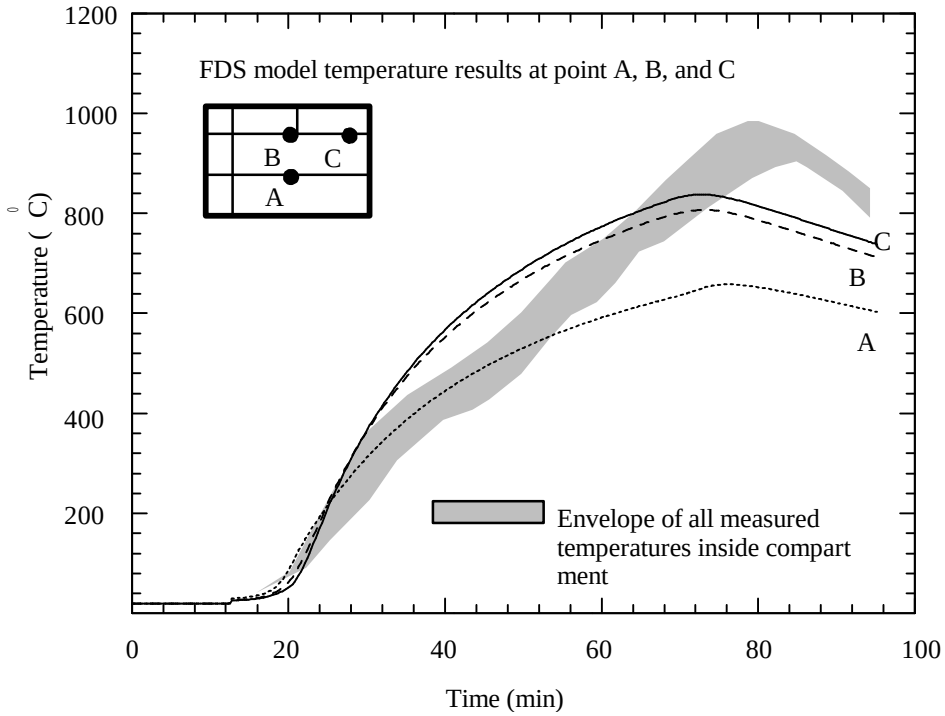


Figure 6-21. Predicted temperature histories compared with experimental values for different beams in the third Cardington fire test.

6.6 Transient heat model for the slab in the third Cardington test

A separate heat transfer analysis for the third Cardington test is also performed to get the temperature profiles for the concrete slab. The transient heat analysis is modeled using four-node quadrilateral shell elements (DS4). Similar to the fourth Cardington fire test, Figure 6-22 shows the FE model for the concrete and the read area is used for the third Cardington fire test. The number of element and nodes used in the model are 6372 and 7400, respectively. The FDS heat flux results are applied as boundary conditions to the bottom surface of the slab while a room temperature is employed to the top surface. The concrete section temperature results from the heat transfer analysis are compared with the experimental data at the CS1 location (Figure 6-19). Figure 6-23, 6-24, and 6-25 show comparisons between the experimental and model results. Good overall correlation is demonstrated. In addition, the results from a separate analysis of Lamont et al. [39] are also shown in these Figures. Lamont et al. [39] uses a program called HADAPT for their 2D nonlinear transient thermal analysis. A uniform temperature history obtained from the experimental data for the surface of the concrete slab is used in their analysis model, instead the spatial temperature history profiles from the fire simulation using FDS model are applied to our heat transfer analysis to generate the temperature in the concrete slab.

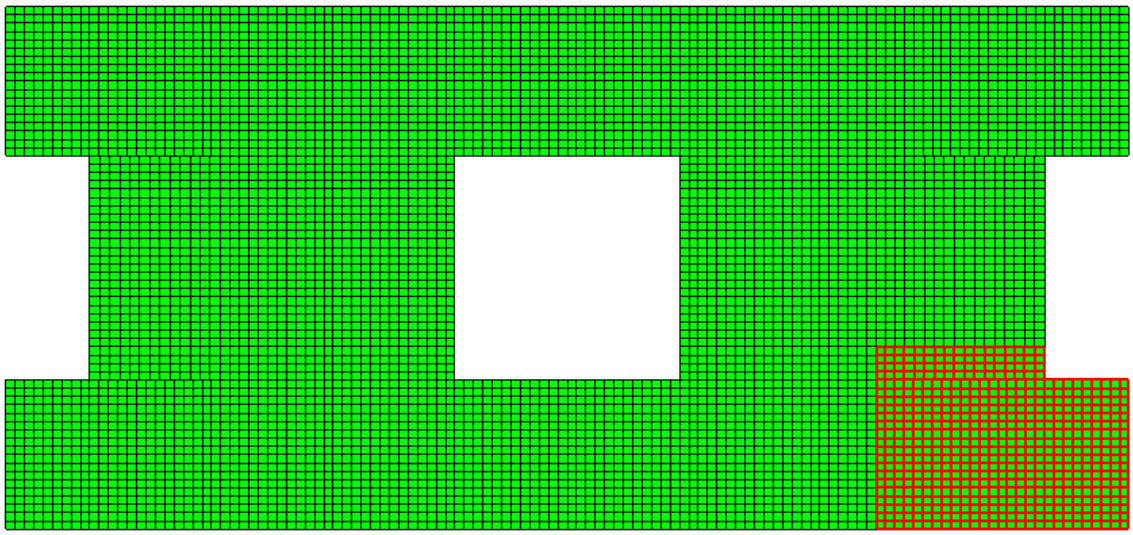


Figure 6-22. Heat transfer analysis FE model for the third Cardington test.

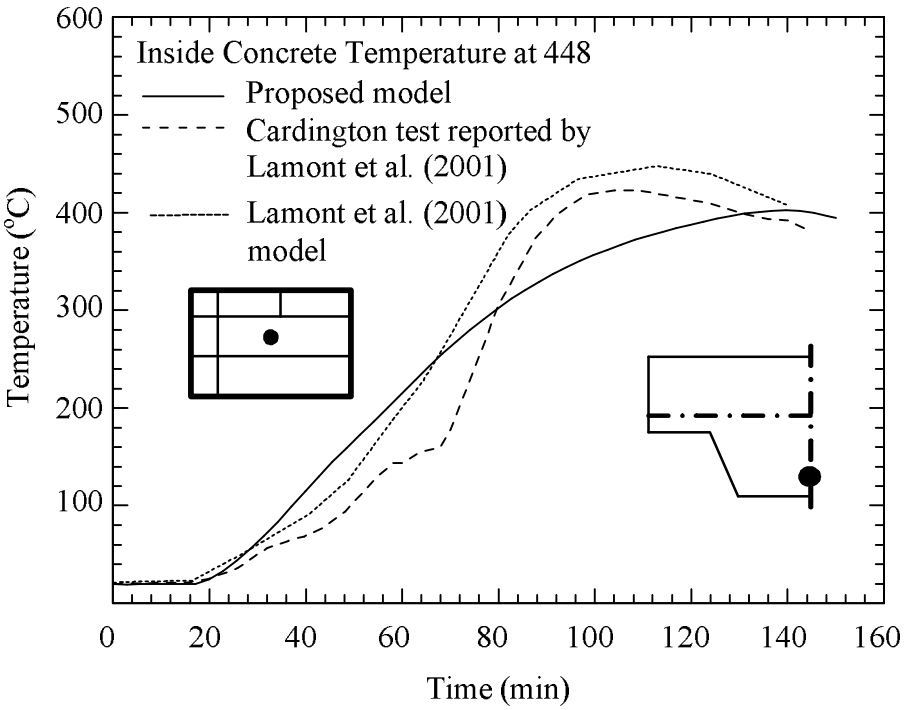


Figure 6-23. Predicted temperature history at a point (448) in the concrete slab from heat transfer analysis of the proposed model compared with the reported experimental data and the model proposed by Lamont et al. [39].

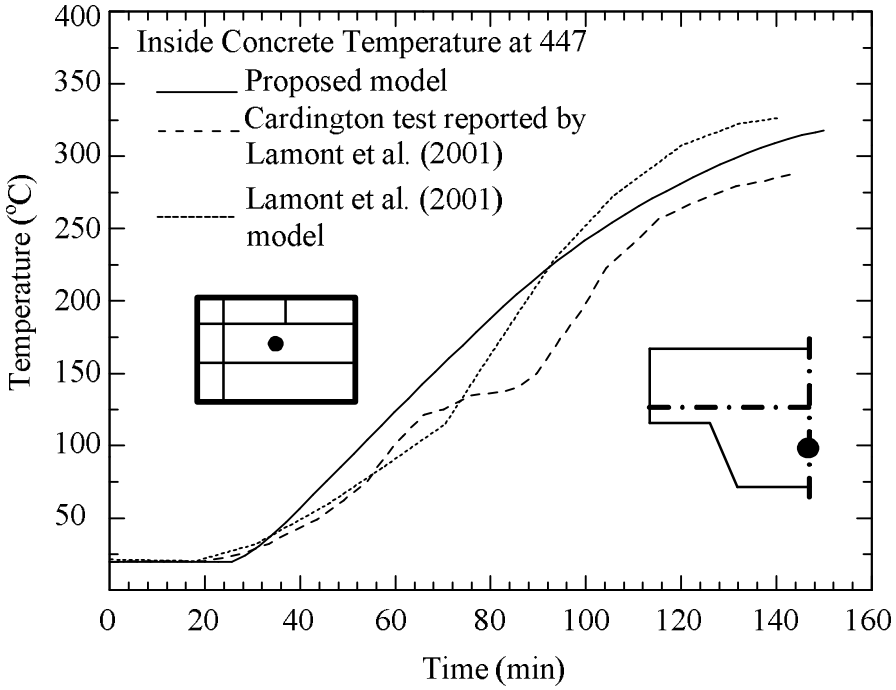


Figure 6-24. Predicted temperature history at a point (447) in the concrete slab from heat transfer analysis of the proposed model compared with the reported experimental data and the model proposed by Lamont et al. [39].

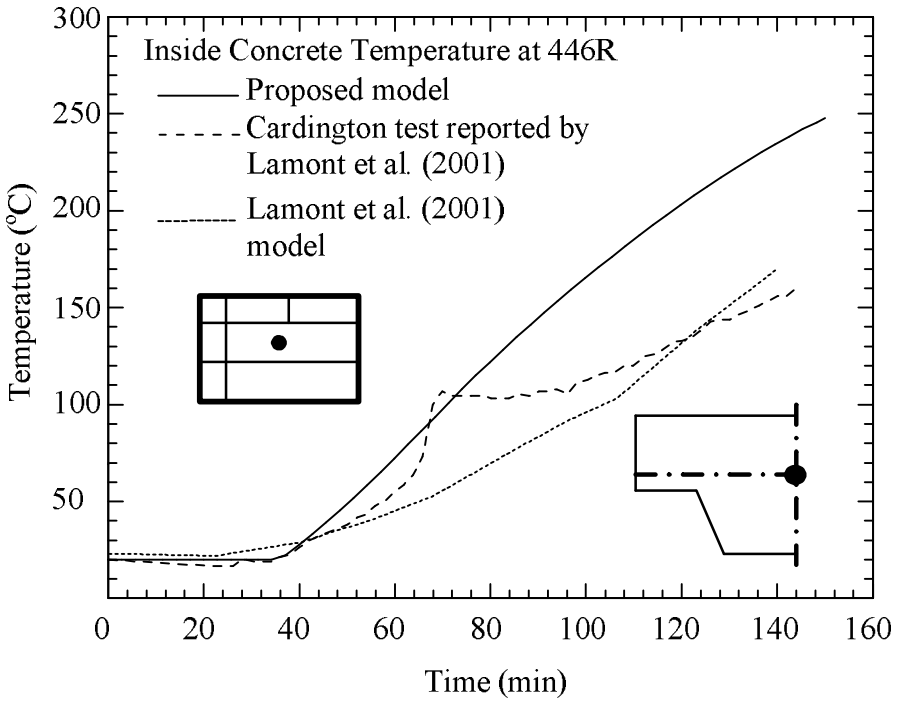


Figure 6-25. Predicted temperature history at a point (446R) in the concrete slab from heat transfer analysis of the proposed model compared with the reported experimental data and the model proposed by Lamont et al. [39].

6.7 Concurrent nonlinear thermal structural analysis of the third Cardington test

The thermomechanical FE model for the Cardington test consists of 3D beam elements (Timoshenko beam theory, B31) used for the beams and columns. Four-node shell elements (S4R) are used for the concrete slabs. Figure 6-26 shows the stress analysis FE model used for the third Cardington fire test. The stress analysis model is used for both the third and fourth Cardington fire test. Only the locations of the tests are different. Same as the fourth test, the half of the concrete slab is considered and the red rectangular represent the area of the third Cardington test. Number of elements and nodes used in the stress analysis model are 18441 and 42154, respectively. The temperature history is applied at all nodes of the beam elements and taken from the previous FDS models. The heat transfer analysis provides the temperature history for the shell elements representing the concrete slabs. The predicted vertical deflections are

compared with the Cardington experiments reported by Usmani et al. [57] at several points. Figure 6-19(b) illustrates the locations where deflections are measured. Figures 6-27 to 6-34 show the predicted deflections for eight locations compared with experiments and a separate structural model reported by Usmani et al. [57]. Both models, the current and Usmani et al. [57], shows relatively good prediction abilities compared with the test data. This highlights the need for including nonlinear concrete damage model as part of the analysis. The deflection results of Usmani et al. [57] model have been limited to pre-peak temperature history, i.e. the portion of the fire history less than 100 min. Both models show more difference with the experiment results towards the end of the fire where maximum damage and temperature occurred. However, the current model does include the cooling down which can be critical from nonlinear and damage in structural behavior.

Gillie et al. [25] proposes a modified model to the Usmani et al. [57]. The recent results show very good correlation to the deflection-temperature relations. However, it is not clear how the new analysis pairs when compared to the temporal deflections since we could not represent their temporal deflections from the published results. The motivation for presenting the structural response in terms of deflection-temperature relations is not clear and justified since this representation cannot be used for post-peak cool down history; the temperature function is decreasing and not unique. Their model use time-dependent temperatures measured from the test and applied uniformly for the concrete slab. Similar approach is reported for the beams. The current analysis is performed using general purposed four-node reduced integration shell element for concrete slab. Time and location-dependent (spatial-temporal) temperatures predicted from FDS and heat transient analysis are applied for thermal loadings. Slab temperatures in the compartment fire, such as the third Cardington fire test, can be spatially heterogeneous and non-uniform. The proposed integrated frame work can capture this spatial dependency and apply it to the structure in order to capture localized behavior and effects.

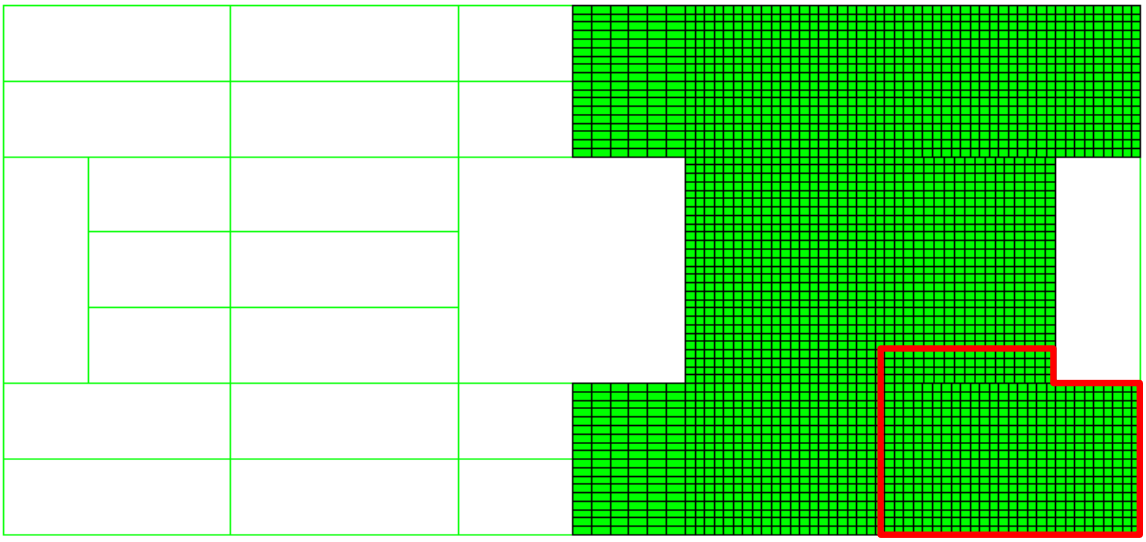


Figure 6-26. Stress analysis FE model for the third Cardington fire test (top view)

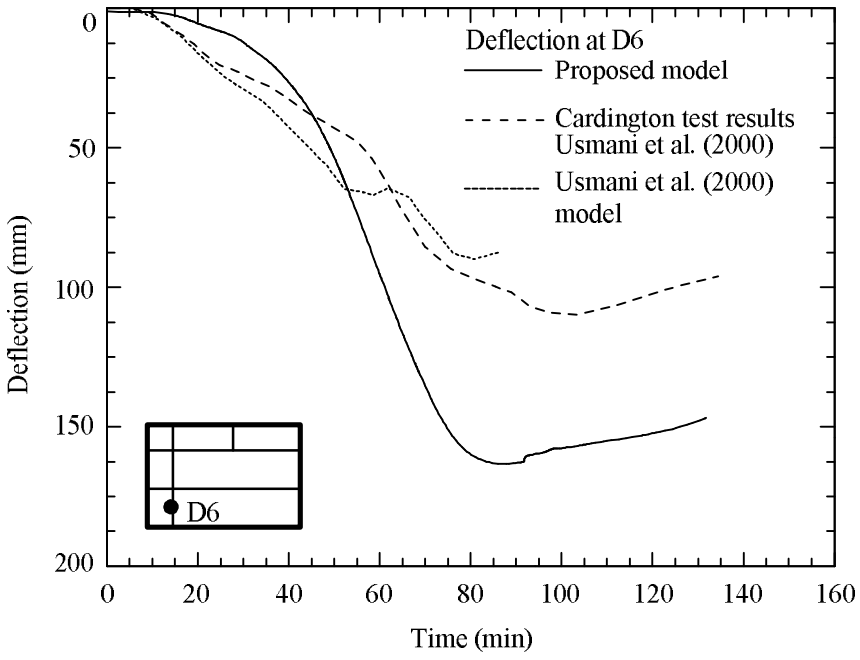


Figure 6-27. Predicted beam deflection history (at the circled point, D6) compared with the reported experimental data and the model proposed by Usmani et al. [57].

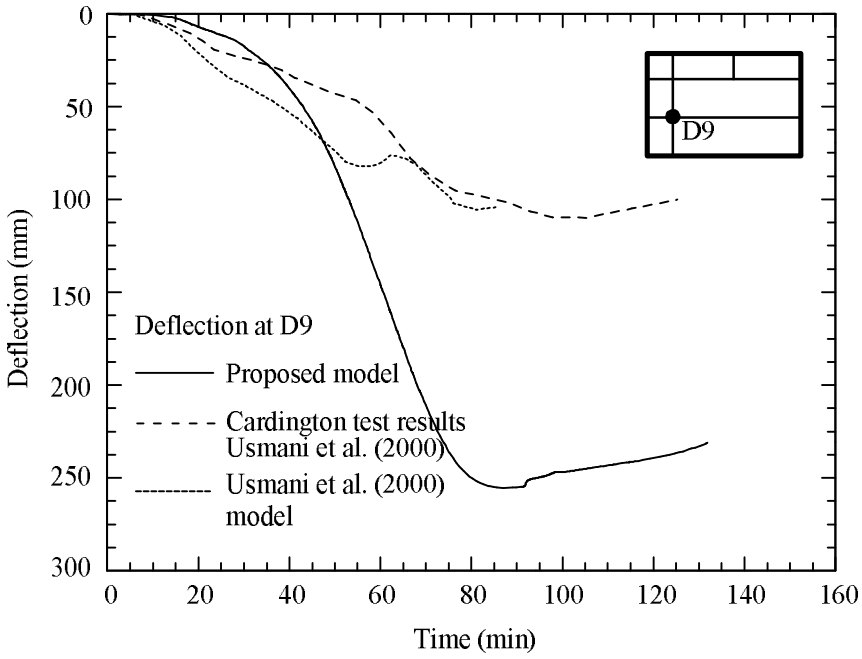


Figure 6-28. Predicted beam deflection history (at the circled point, D9) compared with the reported experimental data and the model proposed by Usmani et al. [57].

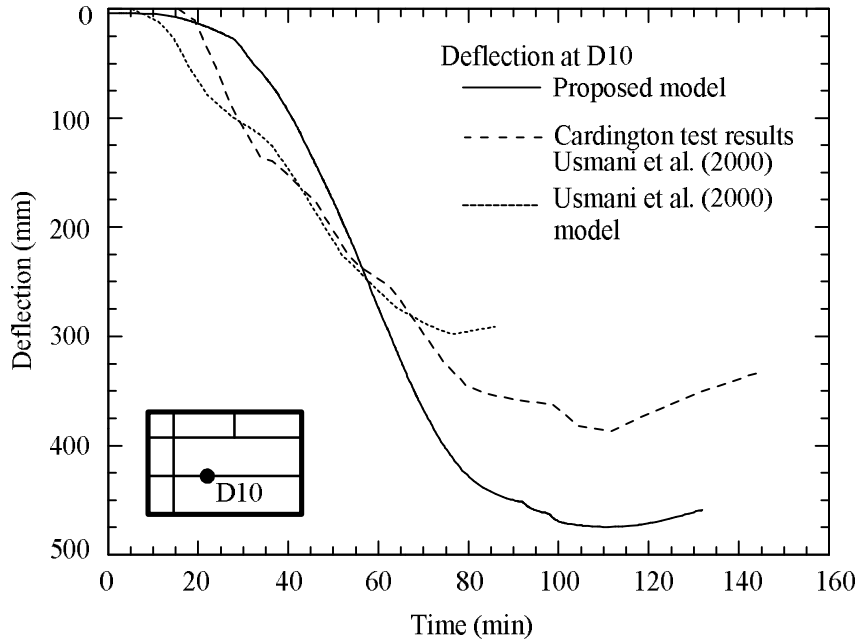


Figure 6-29. Predicted beam deflection history (at the circled point, D10) compared with the reported experimental data and the model proposed by Usmani et al. [57].

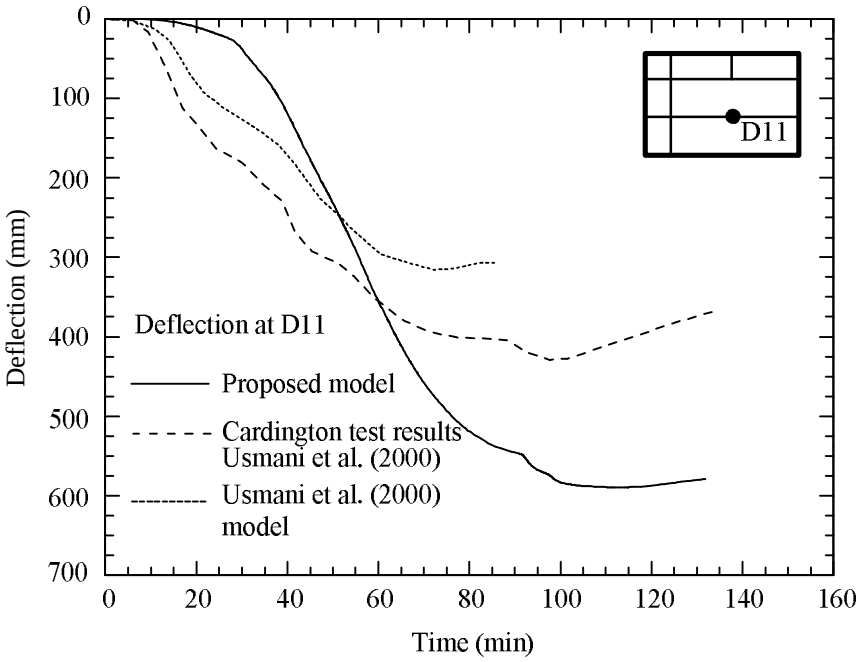


Figure 6-30. Predicted beam deflection history (at the circled point, D11) compared with the reported experimental data and the model proposed by Usmani et al. [57].

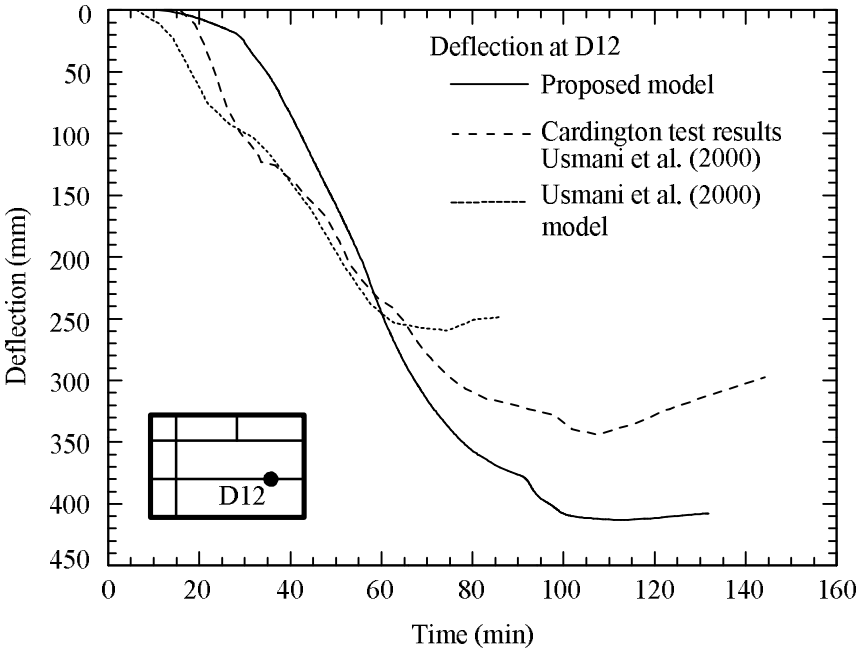


Figure 6-31. Predicted beam deflection history (at the circled point, D12) compared with the reported experimental data and the model proposed by Usmani et al. [57].

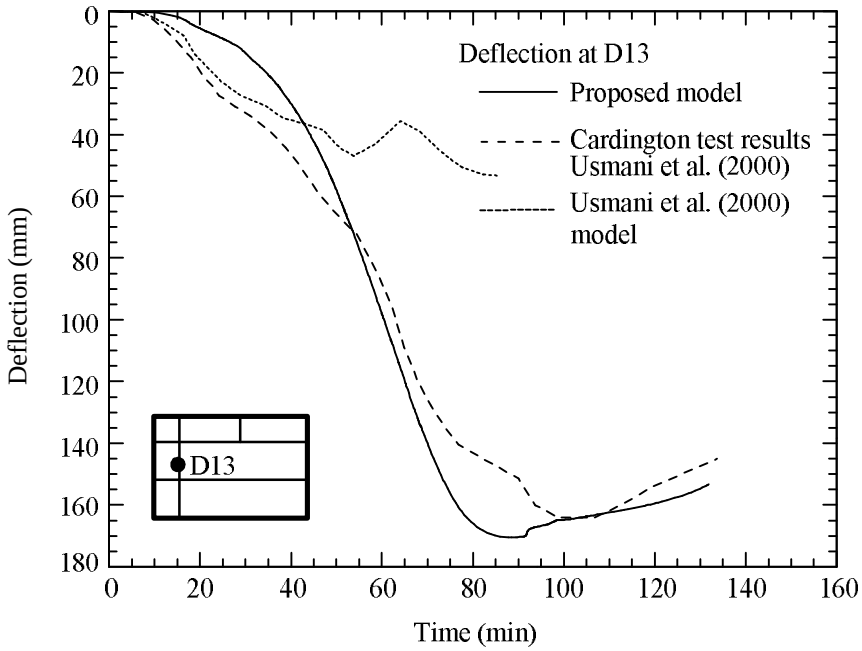


Figure 6-32. Predicted beam deflection history (at the circled point, D13) compared with the reported experimental data and the model proposed by Usmani et al. [57].

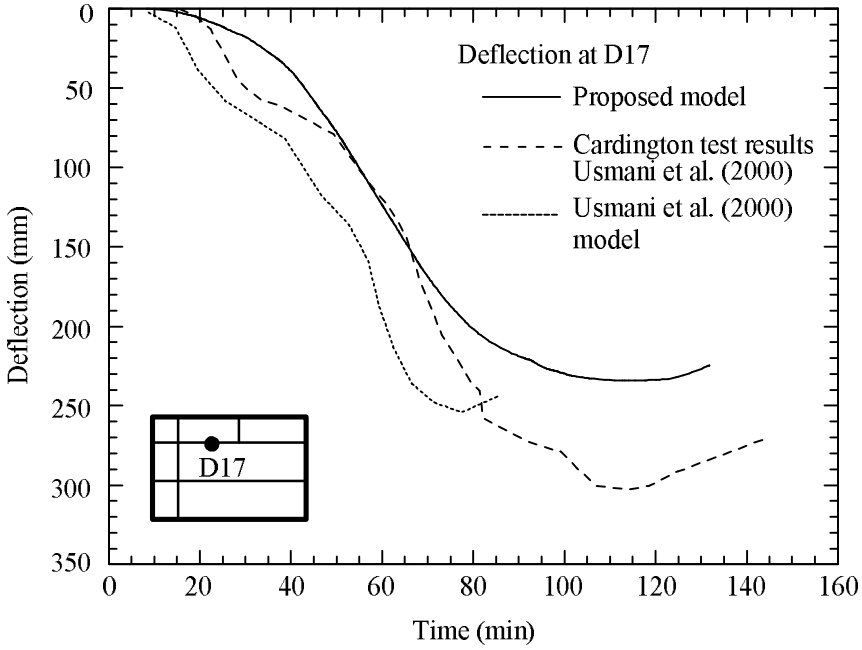


Figure 6-33. Predicted beam deflection history (at the circled point, D17) compared with the reported experimental data and the model proposed by Usmani et al. [57].

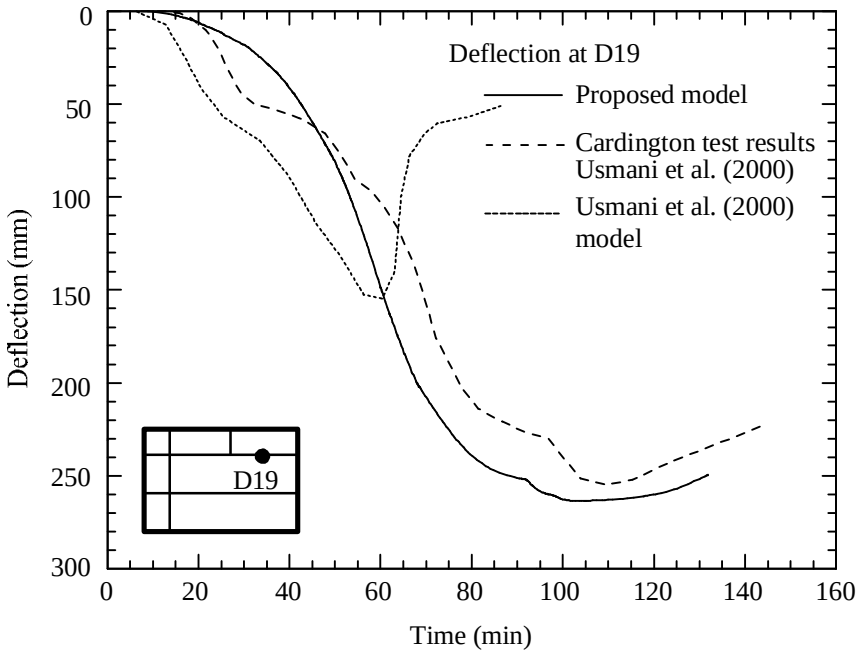
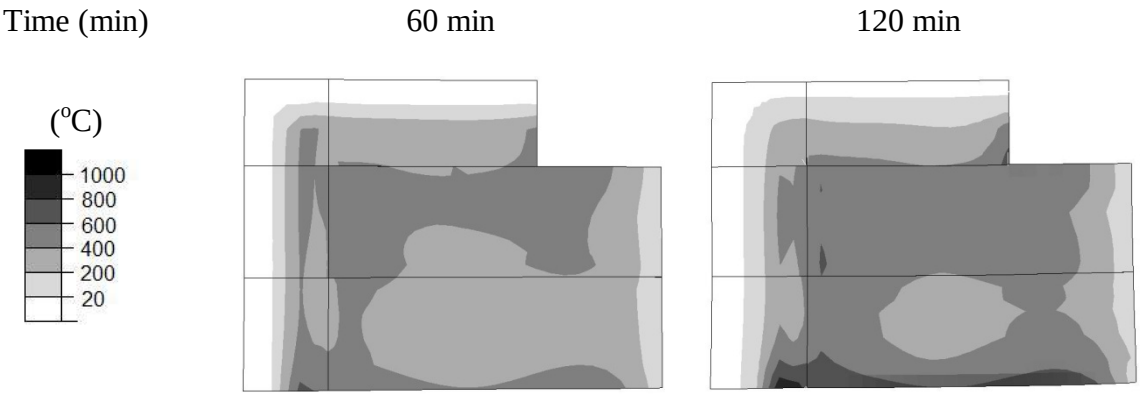


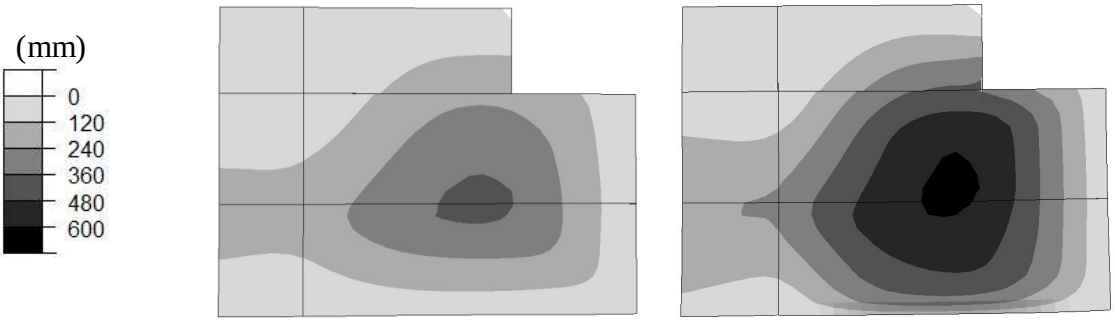
Figure 6-34. Predicted beam deflection history (at the circled point, D19) compared with the reported experimental data and the model proposed by Usmani et al. [57].

Next, we illustrate the importance of performing full-scale fire simulation prior or concurrently with the structural analysis by showing the spatial variation of the temperature and deformations. Figure 6-35 shows a table with the non-uniform temperature, deflection, and stresses distributions of the bottom of the concrete slab obtained from the proposed model at 60 and 120 minutes. The test compartment begins to cool down around 100 minutes after the fire consumed all fire sources as shown in Figure 6-20. The contour plots at 60 minutes and 120 minutes present the states during heating and cooling, respectively. The concrete slab temperatures are still increasing, even though the steel beam and air temperatures are going down during the cooling stage. This concrete temperature increment is because the air temperature inside test compartment is relatively higher than the temperature of the concrete slab during this stage. The stresses at the bottom of the concrete slab around steel beam are also increased in compression because of the relatively big temperature differences between the concrete slab and steel beams. These figures clearly illustrate the importance of

performing a full-scale fire simulation to accurately predict the spatial temperature distributions. In addition, the new results highlight the importance of performing the structural analysis for both heating and cooling fire stages as in the later deformation gradients may also be critical due to damage and geometrical constraints. All detailed figures shown in the Figure 6-35 are provided in the appendix A.1.

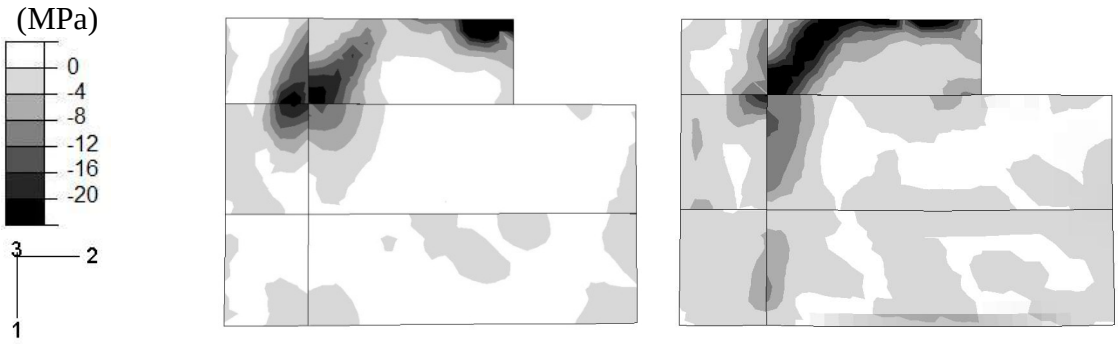


(a) Predicted temperature contours at bottom of concrete slab

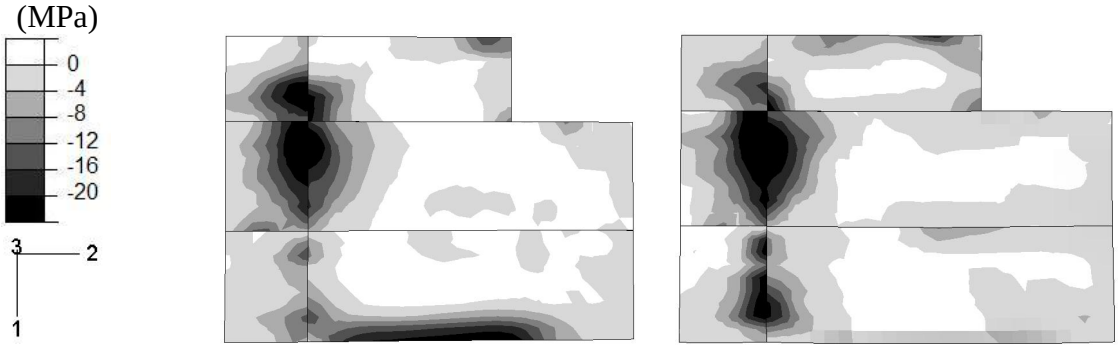


(b) Predicted vertical deflection contours

Figure 6-35. Predicted spatial distribution of temperature, deflection, and stress contours before and during cooling. (topview) (continue to the next page)



(c) Predicted stress contours for 1-direction at bottom of concrete slab



(d) Predicted stress contours for 2-direction at bottom of concrete slab

Figure 6-35. (Continued) Predicted spatial distribution of temperature, deflection, and stress contours before and during cooling. (topview)

CHAPTER 7

BRIDGE MODELING UNDER FIRE: INTEGRATED FIRE DYNAMICS AND THERMOMECHANICAL FRAMEWORK

The proposed integrated fire dynamics and thermomechanical modeling approach is used to present an efficient and refined structural analysis of bridges under fire loading. The framework is applied to different steel and concrete bridge structural systems in this chapter. The objectives of this chapter are to show the applicability of the proposed analysis framework to predict the behavior of the bridge during fire situations. In addition, the analysis framework can be used to evaluate the bridge structure damage levels after fire. In the following, the thermomechanical analysis framework is applied to few well-known major fires that occurred under major bridge systems. Some of these cases are not well-documented, and the fire case was not possible to obtain from the scientific literature. In this case, some assumptions were used in part based on media reports of these accidents, such as their duration, the location of the fire, time to collapse, and the amount of fuel or fire source. Having said that, the overall objective of this Chapter is to demonstrate the applicability of the proposed framework to large-scale bridge systems.

7.1 Fire damage mitigation in bridges and buildings

All structures from basic houses to complex military facilities are in danger of fire or have been subject to fire. The damage from fire is sometimes limited to house furniture or clothes but often major parts of the structure can be damaged which ultimately lead to collapse. There are several ways to reduce fire damage and minimize the risks to the structure and inhabitants. The easiest and simplest method is to extinguish the fire as quick as possible before it grows. The fire alarm system, sprinkler

system, smoke/thermal detectors, and potable fire extinguishers are the fire equipment used for this purpose. However, from a structural engineering point of view, passive fire protection can be constructed and improved using the current analysis tools. The objectives of the passive fire protection in the structure are: to reduce thermal gradients, to try to contain fires, localize damage, and slow their spread. Fire walls, occupancy separations, cable coating, and spray fireproofing are some of the available structural protection techniques. Beside these passive fire protections, there is a design technique to protect the structures from fire damage considering the potential fire sources. Not only the fireproofing coating for steel girders but also the bridge height and kind of vehicle passing are the most consideration to mitigate the fire damage for the bridge structures. The locations of the gas pipes, ventilation systems are also important in the building to reduce the fire risk. The proposed framework makes it feasible to consider the all possible fire damage risks and to investigate their structural damage levels.

7.2 Fuel-truck accident under the Oakland I-580 steel bridge over I-880, CA (April 29, 2007)

A highway bridge overpass in the East Bay's MacArthur Maze in Oakland collapsed on April 29, 2007. Figure 7-1 shows the collapsed highway bridge location from several websites. [71, 72] It is located at the eastbound connector to Interstate-580 overpass the southbound connector Interstate-80 to Interstate -880 in Oakland, California. The collapse occurred due to an accident moving a gasoline truck. A gasoline truck crashed and burst into flames on the southbound connector I-80 to I-880 under the I-580 around 3:40am near the San Francisco Bay Bridge. Because of the heat from the flames, two spans of the I-580 Bridge are collapsed around 4:00 am [64]. It is about 20 minute after the truck is crashed. Figure 7-2 shows the bridge after it is fallen down with different views. A total of 8600 gallons of unleaded gasoline was spilled and caught fire [65]. Everyday 280,000 commuters take the bridge into San Francisco. The bridge was

re-opened on May 25, 2007 in the morning. The replacement of the steel girders and bridge decks took 26 days.

(a) Google maps from http://maps.google.com/ [71]

(b) Live Search maps from http://maps.live.com/ [72]

Figure 7-1. Oakland I-580 Bridge over I-880

(a) photo by Lacy Atkins

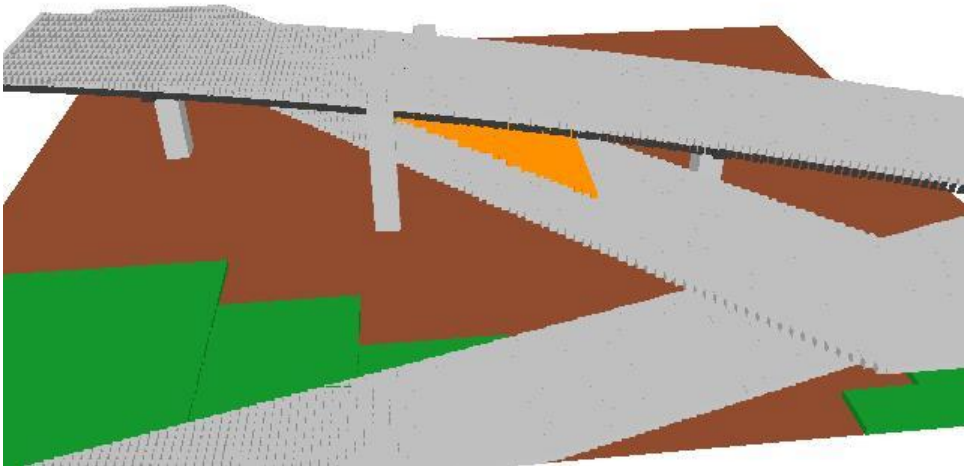
(b) photo by Robert Campbell

Photos from SFGate.com (http://www.sfgate.com/cgi-bin/object/article?f=/c/a/2007/04/29/BAGVOPHQU46.DTL&o=) [75]

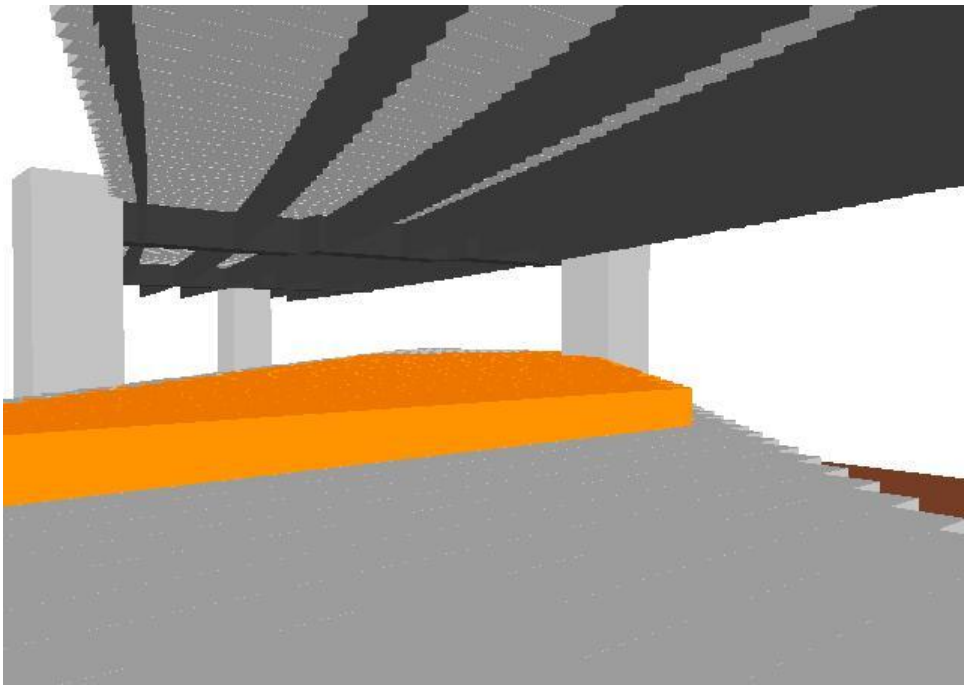
Figure 7-2. The collapsed Oakland bridge (eastbound connector to I-580)

7.3 Proposed fire simulation model for the Oakland-San Francisco Bay Bridge

We develop a FDS model for the collapsed I-580 bridge including I-80/880 bridge where the gasoline truck crashed. Figure 7-3 shows the FDS bridge model with overall and on the I-80/880 bridge deck view. We approximate the bridge geometry based on Google satellite maps [71] because the exact bridge dimensions are not available to us. The assumed width and height of the bridge are as 14 and 10 meters respectively. The spilled gasoline from the crashed truck is shown in Figure 7-3 as orange color objects. Ninety percent of the total spilled gasoline (8600 gallons) is on the I-80/880 bridge deck and other 10% of gasoline is on the ground because we assume that most of the spilled gasoline is on the bridge but also some of it flowed to the ground. Heat release rate per unit area for the fire source is assumed as 2500.0 kW/m^2 during 21 minutes shown in Figure 7-5. It is assumed that the fire is ignited very quickly and then suddenly approached to steady-state phase. The concrete material is used for bridge decks and columns and steel is for bridge girders. The gray and black objects represent concrete and steel. Wind is also considered as 2.6 m/s NW [66] in our FDS model. Figure 7-4 shows the surface temperature FDS results at the end of simulation when the time is at 21 minute after fire started. The temperatures above the spilled gasoline are around 1000 ^oC in the figures. The surface heat flux and temperature results from the FDS simulation will be used in the nonlinear transient heat transfer and 3D thermal stress analyses.

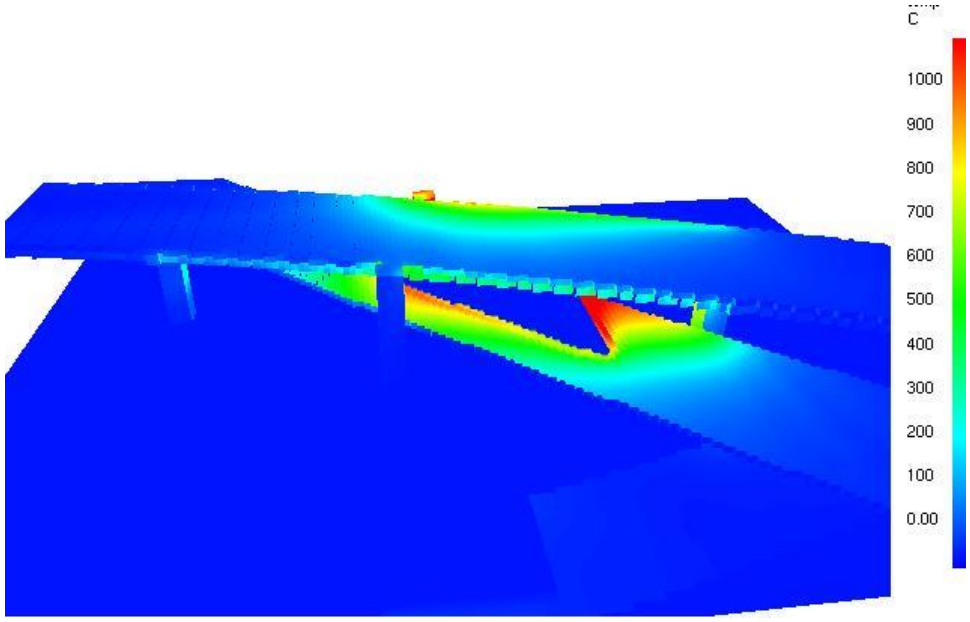


(a) outside view of the bridge

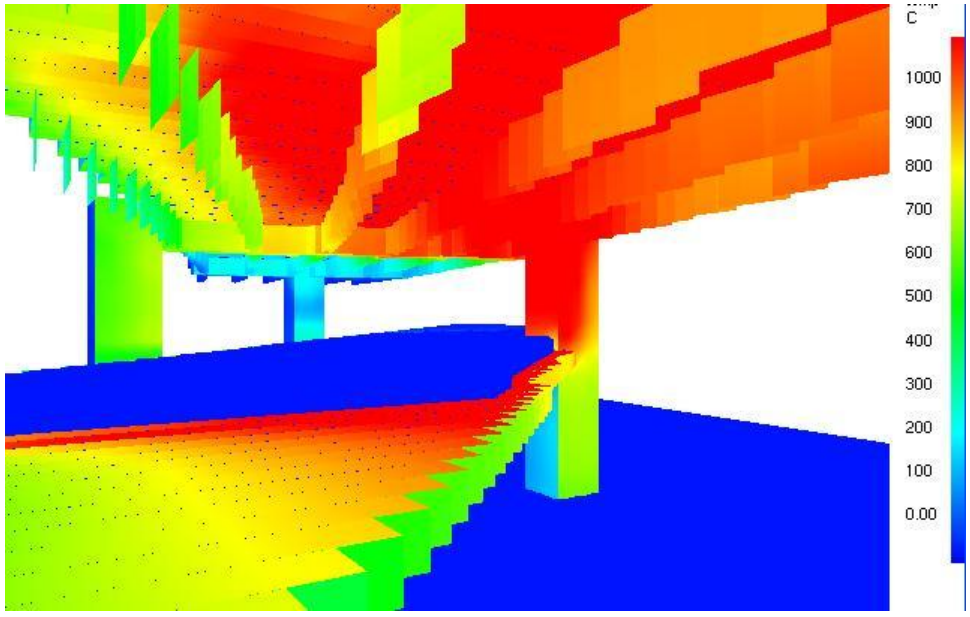


(b) view under the I-580

Figure 7-3. The collapsed Oakland bridge FDS model



(a) outside view of the bridge



(b) view under the I-580

Figure 7-4. The collapsed Oakland bridge FDS model with temperature contour results

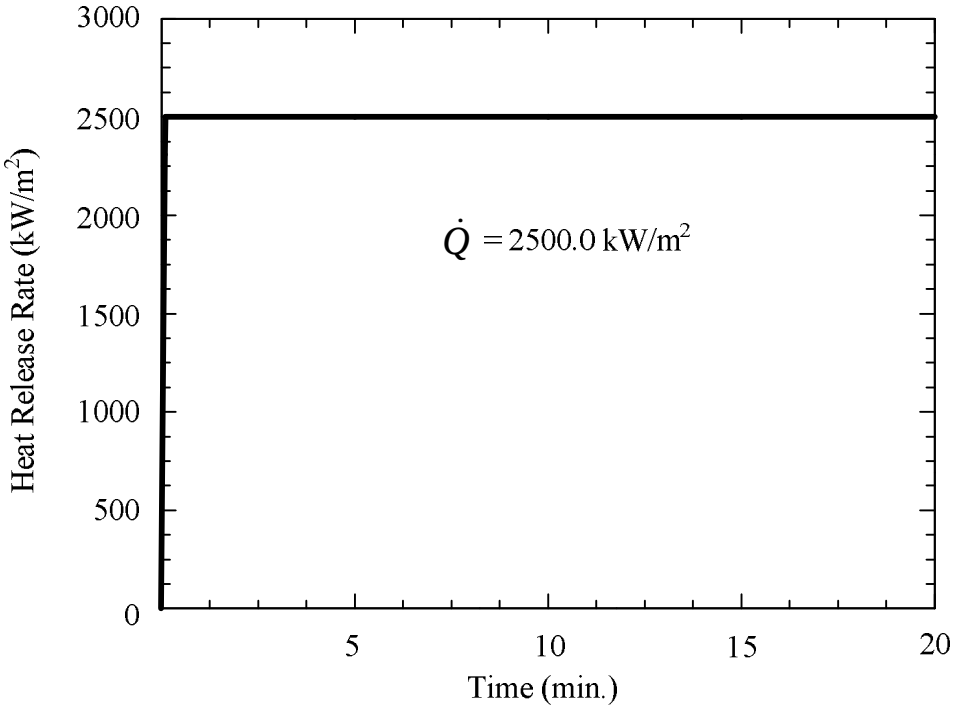


Figure 7-5. Fire load used to simulate FDS source in the collapsed Oakland bridge

7.4 Nonlinear transient heat and thermomechanical FE model

A nonlinear heat transfer analysis for the Oakland Bridge is also performed to get the temperature profiles for the concrete deck. Similar to the previous Cardington fire test simulations, the nonlinear transient heat analysis is modeled using four-node quadrilateral shell elements(DS4) and the concrete bridge deck is only considered in this model. Figure 7-6 shows the heat transfer analysis FE model. Number of elements and nodes are 3344 and 3471, respectively. The FDS heat flux results are applied as boundary conditions to the bottom surface of the deck while a room temperature is employed to the top surface. Figure 7-8 presents the positions through the concrete deck where the temperatures are predicted and Figure 7-7 shows the locations where FE predictions of temperatures and vertical displacements. Nonlinear FE transient heat prediction temperature results for the concrete bridge deck at location D3 are shown in Figure 7-9. The temperature at bottom and middle of the deck is shown using a solid line

(N1) and dash line (N3) respectively. The temperature of the concrete bottom deck is increased rapidly prior to 7 minutes and then reached around 1000 ^{o}C at the 20 minutes.

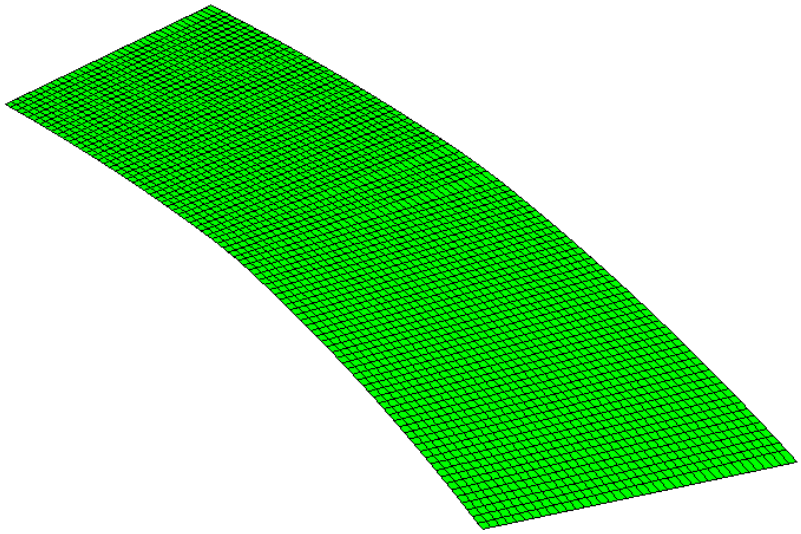


Figure 7-6. Heat transfer analysis FE model for the Oakland Bridge.

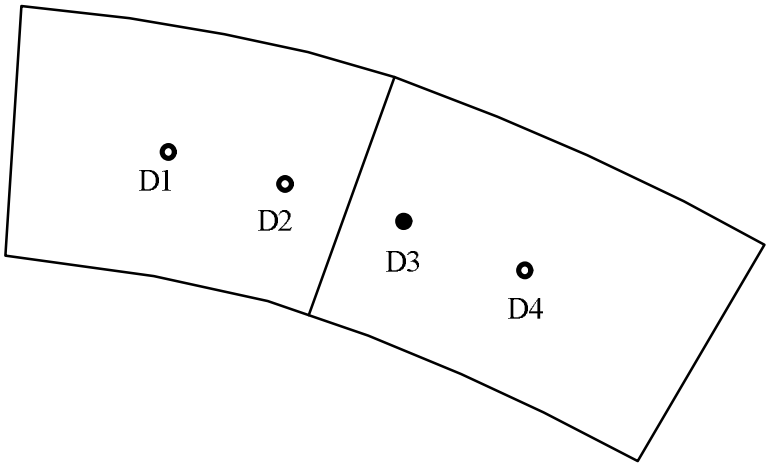


Figure 7-7. Locations where FE predictions for temperatures and displacements

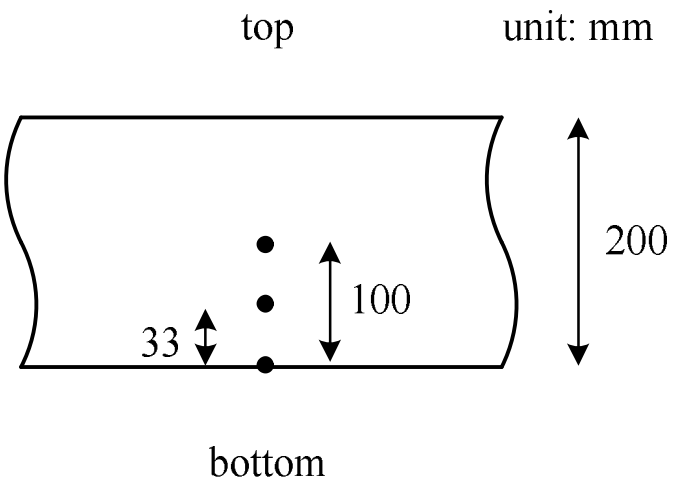


Figure 7-8. Bridge deck FE model cross section

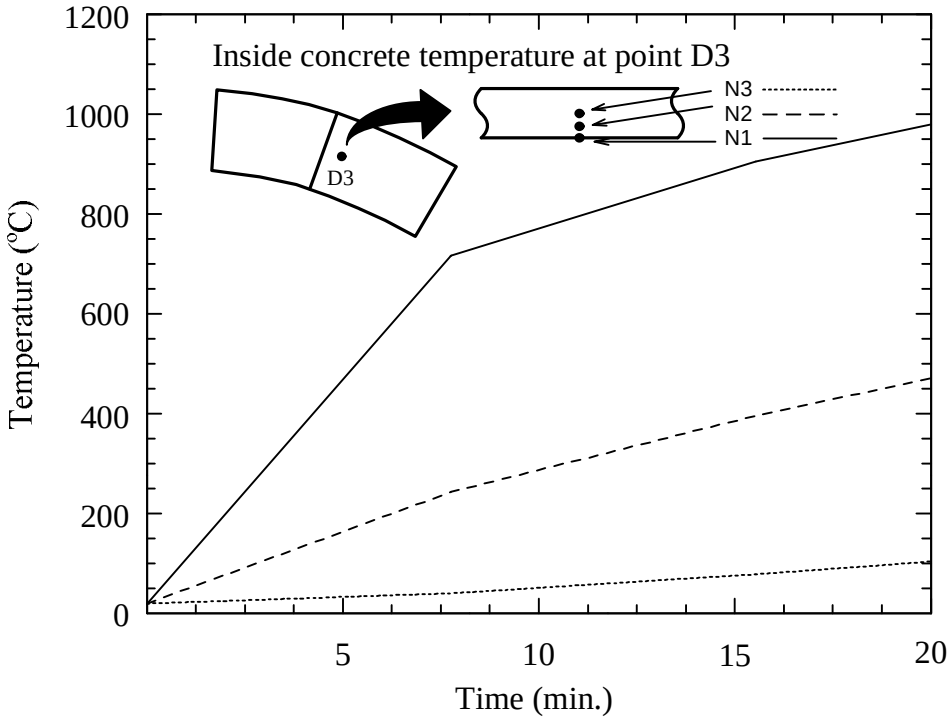
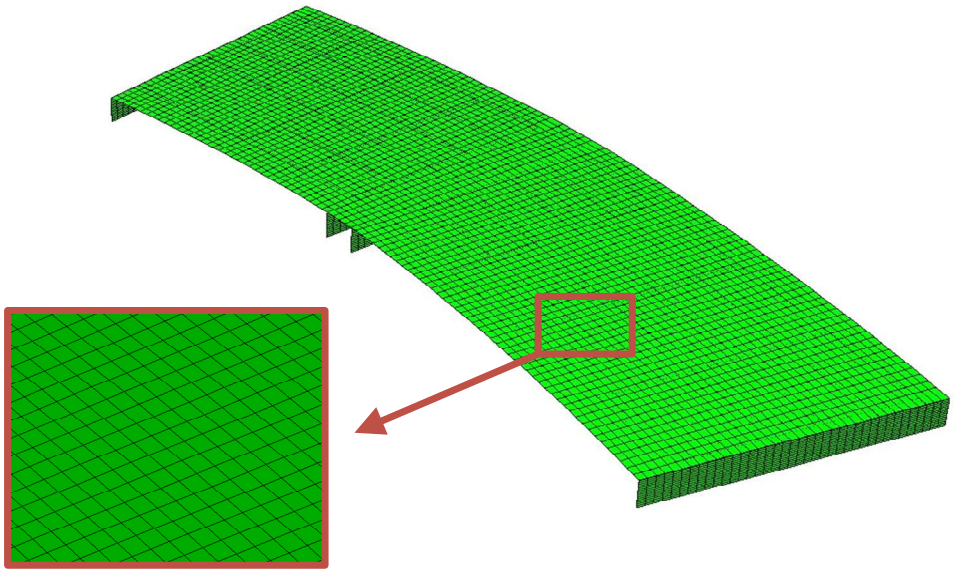


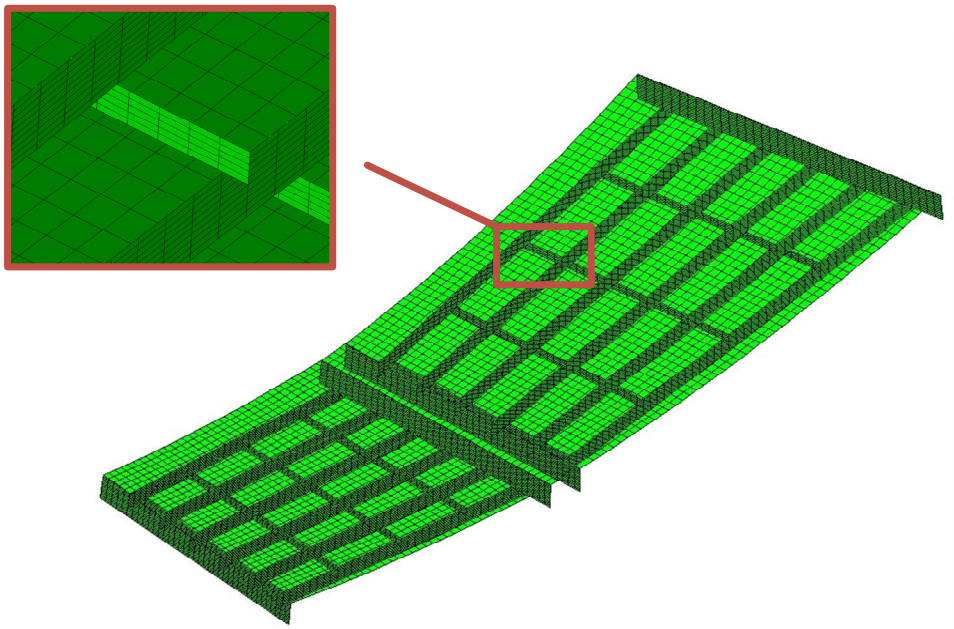
Figure 7-9. Temperature prediction through the concrete deck using heat transfer analysis

The 3D thermomechanical FE model for the Oakland Bridge consists of 3D beam elements (Timoshenko beam theory, B31) used for the flange of the steel girders. Four-node shell elements (S4R) are used for the concrete deck, steel girders' web, and

diaphragms. The temperature histories taken from the previous FDS models are applied to the steel. The heat transfer analysis provides the temperature histories for the concrete bridge deck. Two spans and six steel girders for each span are included in the FE model. Total fifty diaphragms are also modeled. In addition, the concrete deck and steel girders are fully connected. Figure 7-10 shows the 3D thermomechanical FE model. The number of elements and nodes used in the model are 13190 and 15777, respectively. Because the dimensions of the bridge are not available to us, we decide them by several photos in the California Department of Transportation District 4 website [67]. According to the website, we use the following geometry dimensions for our FE model. The thickness of the concrete deck set as 0.2m. Also, 1m, 0.0254m, 0.3m, and 0.0381m are employed for the steel girders' web height, web thickness, flange width, and flange thickness respectively. Self weight of steel girders and concrete deck, parapets, and wearing surface are taken into account for dead load. The 0.48m width by 0.74m height of parapets is assumed to be placed at the both ends of the highway bridge. The wearing surface is put for the last thing considering dead load as 1200 N/m^2. The boundary conditions at the end of bridge spans are considered as simply supported. The concrete deck and the steel girders are fully connected in the model. The temperatures for the steel girders and diaphragms obtained from FDS and applied to the 3D thermomechanical analysis are shown in Figure 7-11 at four different locations. (Figure 7-7) The location D3 is the place where the maximum temperature occurred. It reaches around 1200 ^oC at 20 minute.



(a) mesh of the bridge deck



(b) mesh of the steel girders

Figure 7-10. Collapsed Oakland bridge Stress analysis FE model.

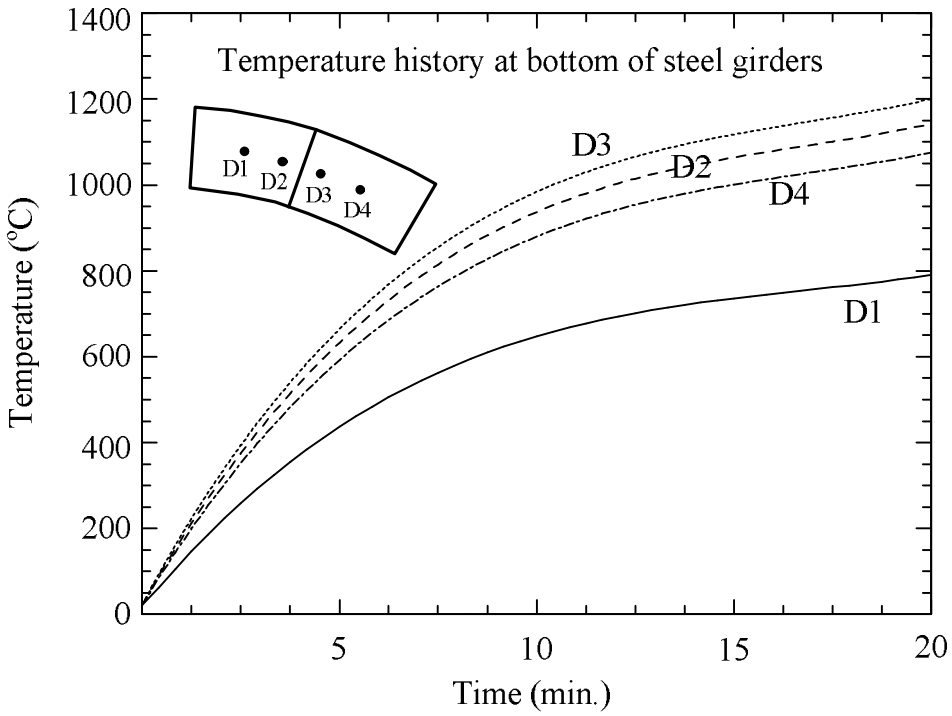


Figure 7-11. The temperatures predicted from FDS and applied to thermomechanical analysis at different locations

The predicted vertical displacements from 3D thermomechanical analysis at four different locations are shown in Figure 7-12 until the time reaches 20min. Relatively large displacements are shown at the middle of the spans (location D1 and D4) and the right span of the bridge (D3, D4). Figure 7-13 shows the bridge pictures of actual video, FDS fire model, and thermomechanical model with three different time step. The actual video shows the bridge just before collapse. According to the equivalent plastic strain contour in Figure 7-13, the steel girders near the middle support are already yielded approximately 15 min after fire occurred. The FE model does not show the collapsed status of the bridge around 20 min but we can predict the behavior based on the equivalent plastic strain and vertical displacement. Around that time, the right span of the bridge shows large deflections and the steel girders near the middle support area have also yielded. The bridge collapse is started at the middle support area of the right span in the actual happening. However, the results using our proposed framework predict the

overall behavior of the bridge under fire conditions. Figure 7-14 to Figure 7-19 show the detailed models at 15min and 20 min after the accident.

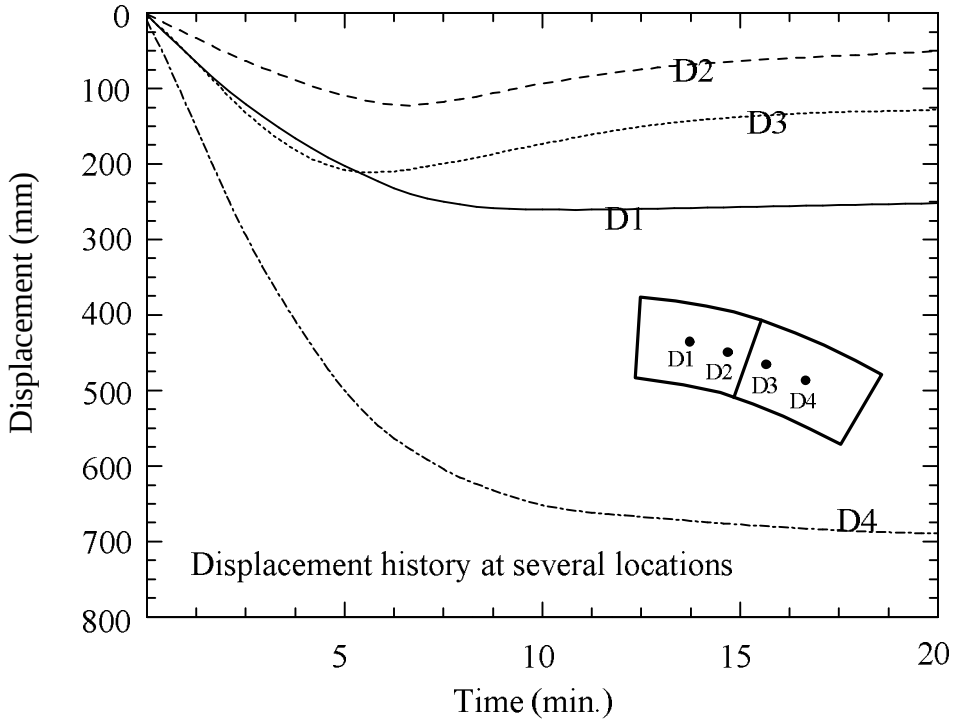
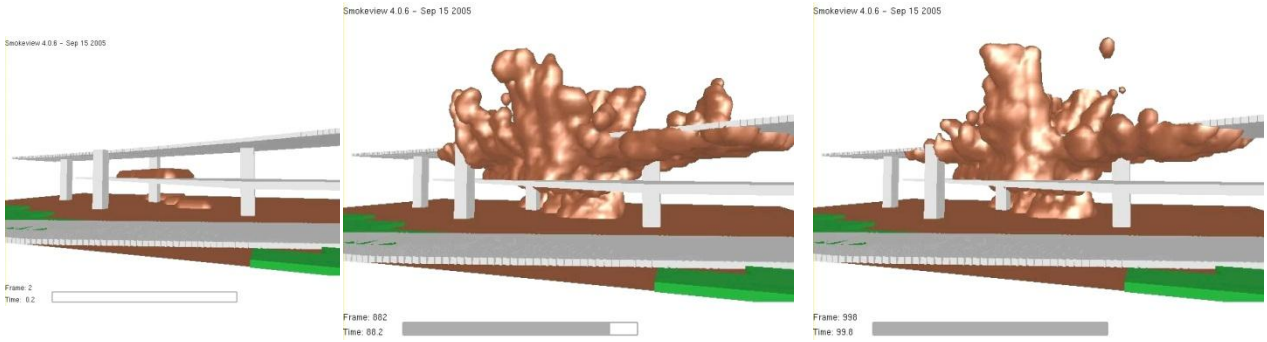


Figure 7-12. Predicted displacements of the collapsed Oakland bridge model at several points

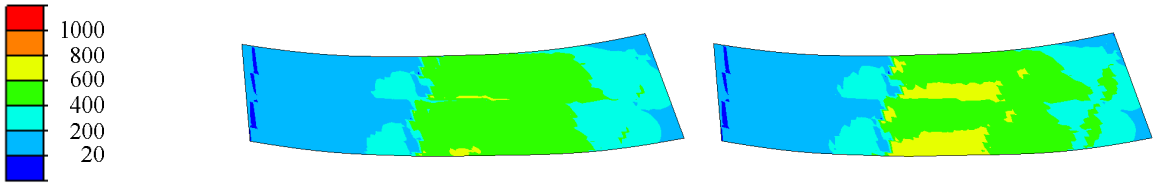
Time (min)
0 min. 15 min. 20 min.

Estimated time from News video

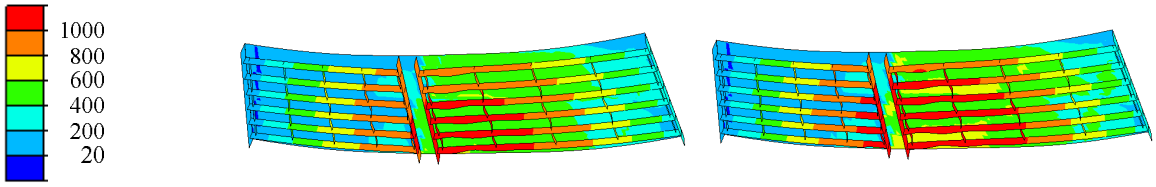
(a) Pictures captured by news reports before and after collapse (http://www.pbs.org/newshour/bb/science/jan-june07/overpass_05-10.html [73])



(b) Predicted heat distribution at estimated times close to the pictures in (a)

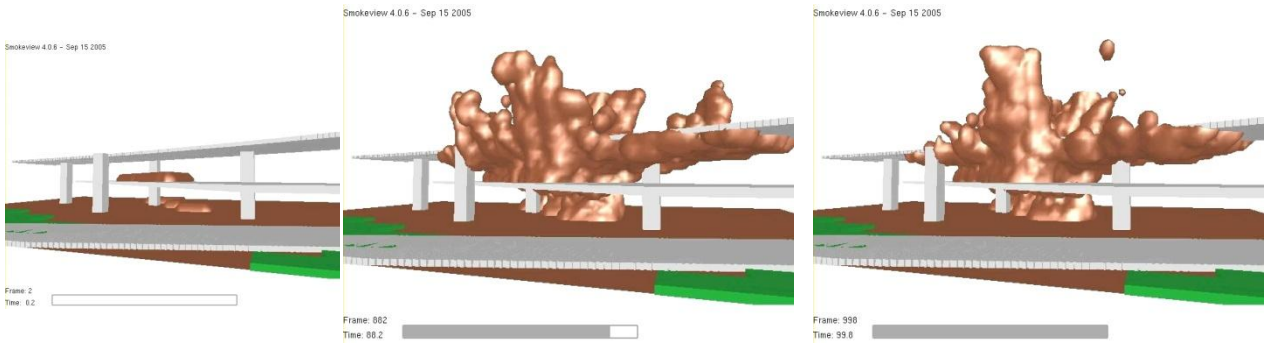


(c) Spatial distribution of temperature of FE heat transfer model for concrete deck at bottom (^{o}C) – bottom view

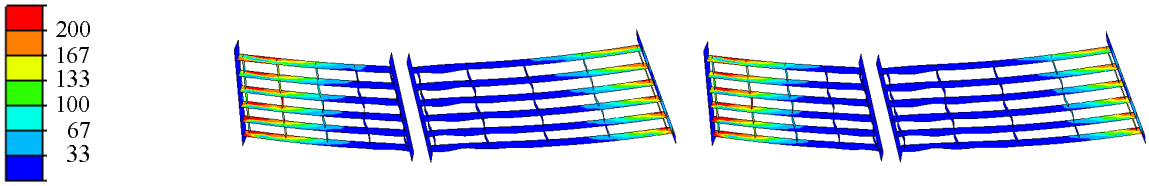


(d) Spatial distribution of temperature of FE structural model (^{o}C) – bottom view

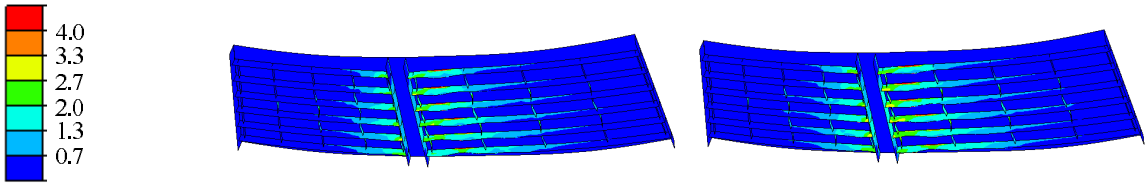
Figure 7-13. Predicted model results for the collapsed Oakland bridge (continue to the next page)



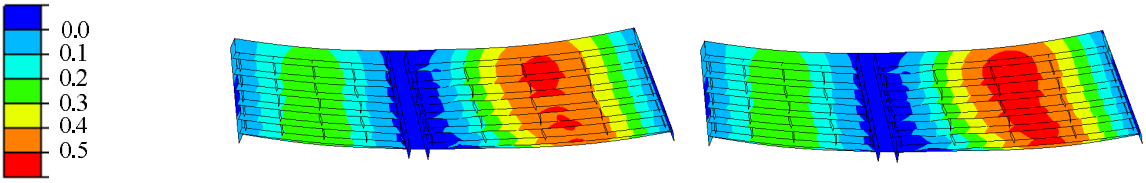
(b) Predicted heat distribution at estimated times close to the pictures in (a)



(e) Spatial distribution of Von mises of FE structural model (MPa) at steel girders – bottom view



(f) Spatial distribution of equivalent plastic strain of FE structural model (%) – bottom view



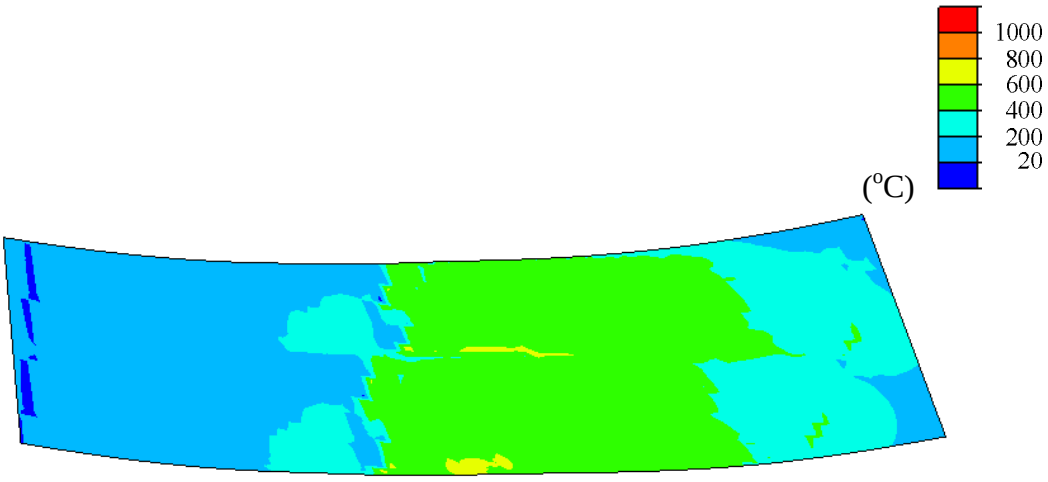
(g) Spatial distribution of vertical displacement of FE structural model (m) – bottom view

Figure 7-13. (Continued) Predicted model results for the collapsed Oakland bridge

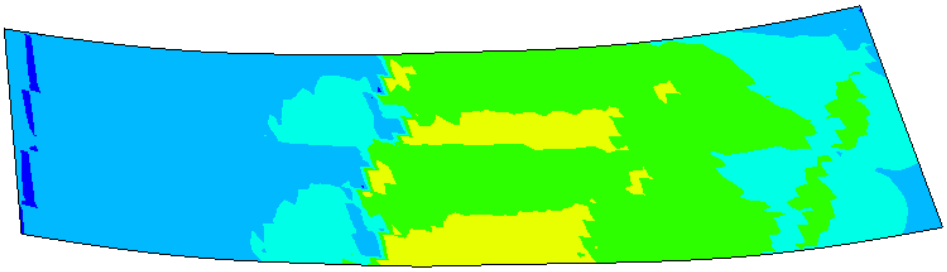
(a) at 15 min after accident

(b) at 20 min after accident

Figure 7-14. Detailed collapsed Oakland bridge FDS model

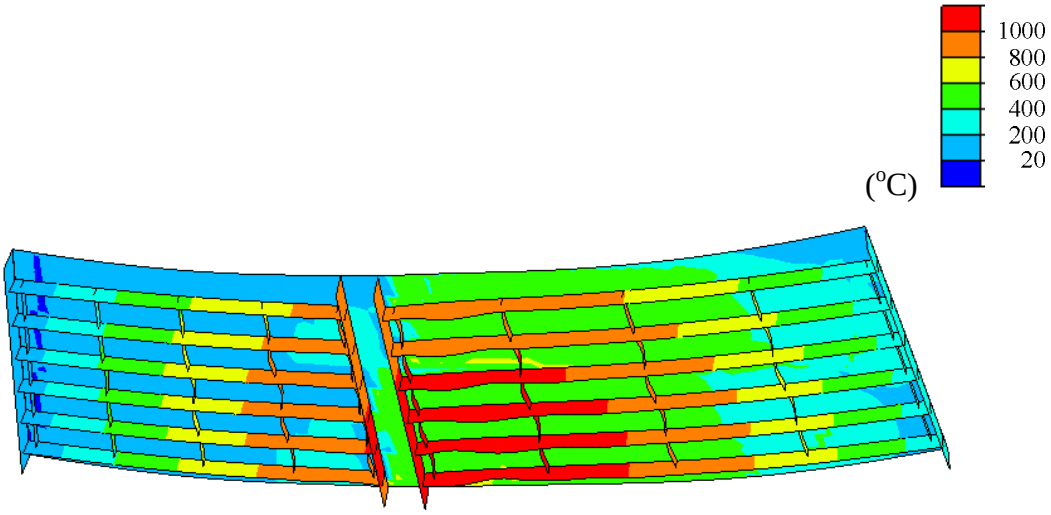


(a) at 15 min after accident

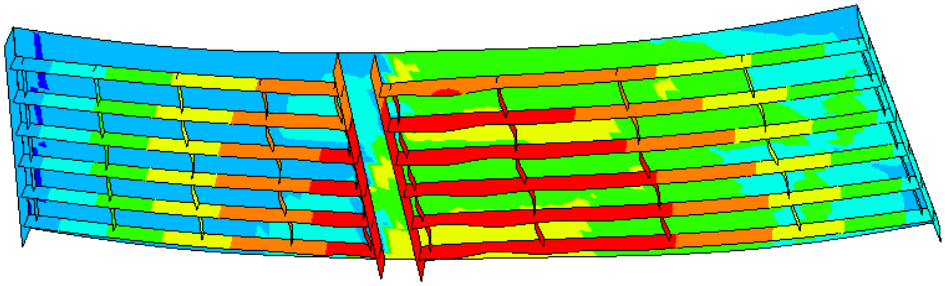


(a) at 20 min after accident

Figure 7-15. Collapsed Oakland bridge heat transfer model with concrete temperature contour

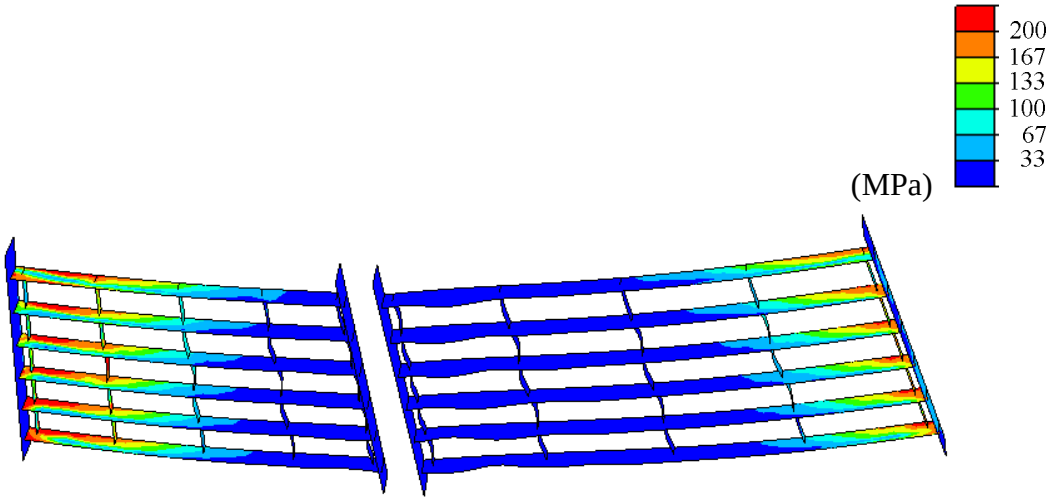


(a) at 15 min after accident

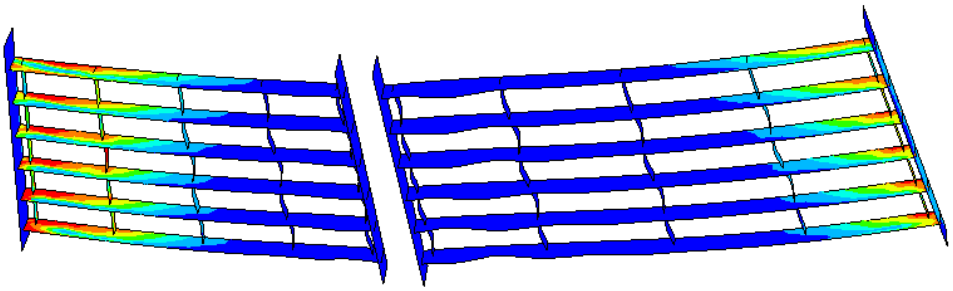


(b) at 20 min after accident

Figure 7-16. Collapsed Oakland bridge thermal stress analysis model with temperature

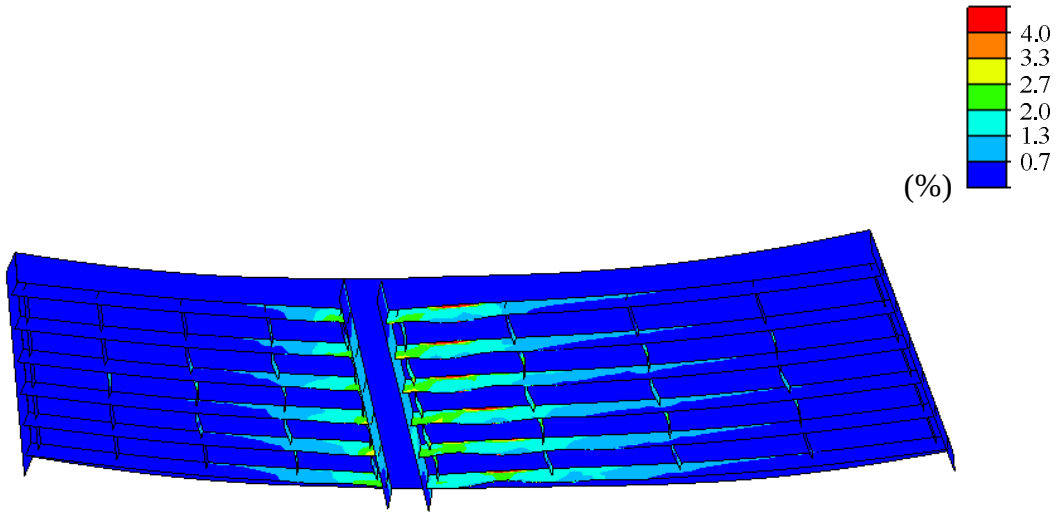


(a) at 15 min after accident

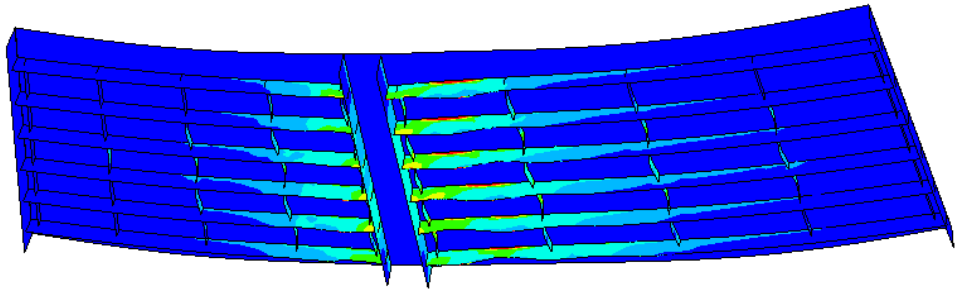


(b) at 20 min after accident

Figure 7-17. Collapsed Oakland bridge thermal stress analysis model with Von mises at steel girders

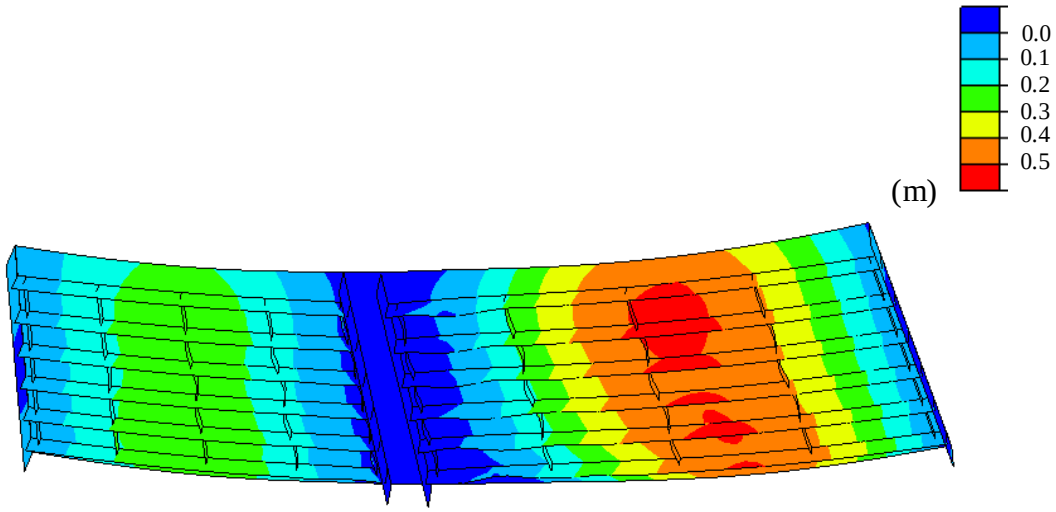


(a) at 15 min after accident



(b) 20 min after accident

Figure 7-18. Collapsed Oakland bridge thermal stress analysis model with Equivalent plastic strain contour



(a) at 15 min after accident

(b) 20 min after accident

Figure 7-19. Collapsed Oakland bridge thermal stress analysis model with vertical displacement contour

7.5 Thermomechanical Parametric Analysis of Coat Layer Fire Damage Mitigation

One of the passive fire protection methods is to coat the structure surface with fireproofing material. The coat layer is able to prevent the surface direct exposure to fire. Simple parametric beam models are performed in this section to demonstrate the concurrent analysis method and study the coat layer thickness needed to mitigate the damage and delay the time to reaching ultimate status.

Finite element models and fire protection coat layer material properties

A straight and a curved beam are simulated with different coat layer thickness. The material properties of the coat layers are assumed to be functions of temperature. Since no complete data is available in the literature about these commercial coat products, this study employs the same temperature-dependence that is well-documented for concrete and apply it to the coat material by scaling the concrete functions with respect to the available room-temperature coat properties. Figure 7-20 and Figure 7-21 show the scaled conductivity and effective specific heat of the coat layer used in the models for the coat in the current section. The conductivity and effective specific heat of the coat layer (at room temperature) are 0.225 W/mK and 0.73 MJ/Km3respectively [74]. The geometry of the simulated beams is 1 meter height, 0.3 meter flange width, and 0.02 meter of thickness of steel. In addition, the total length of the beams is 18 meter and 14 meter for straight and curved beam, respectively. Four-node shell element is used for FE models with both of beam ends are fixed. The beams are subjected to fire loading that is concentrated at their mid-span. The imposed fire heat history in the FDS analysis is shown in Figure 7-22 and a concentrated load is applied to the top flange at their mid-span area.

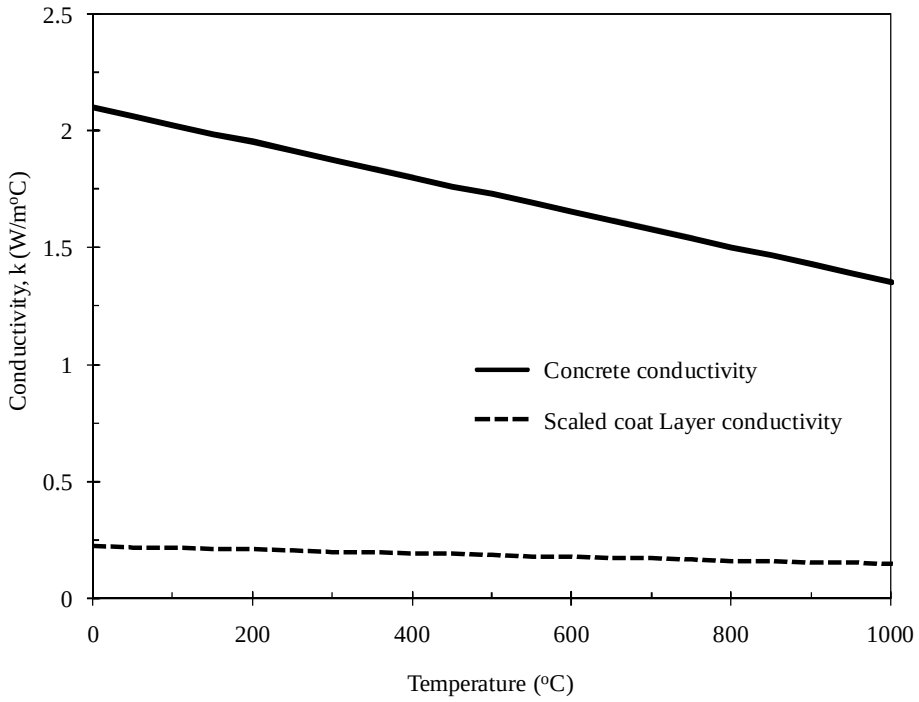


Figure 7-20. Scaled coat material conductivity

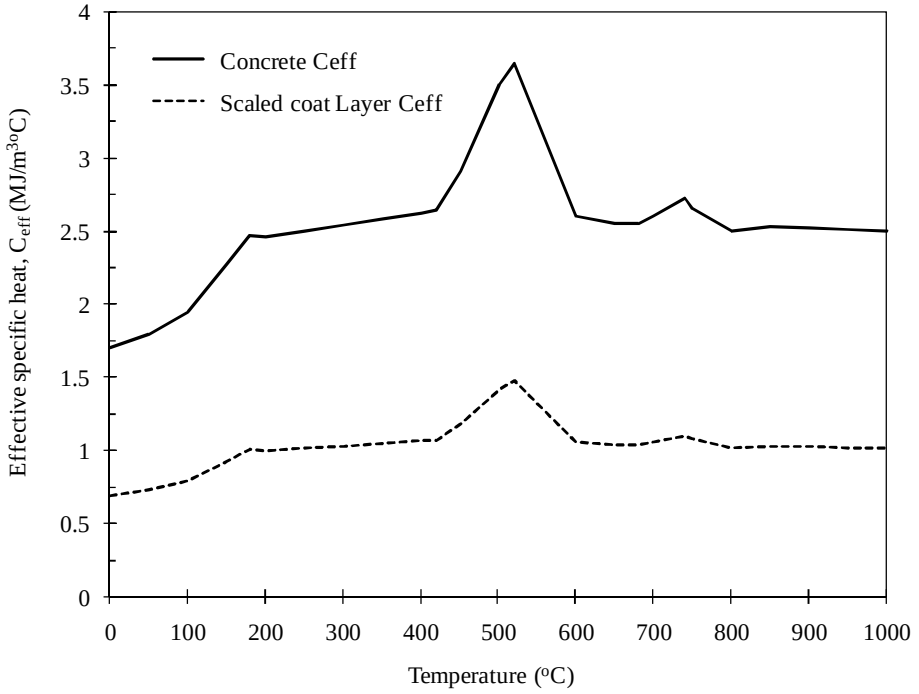


Figure 7-21. Scaled effective specific heat of coat material

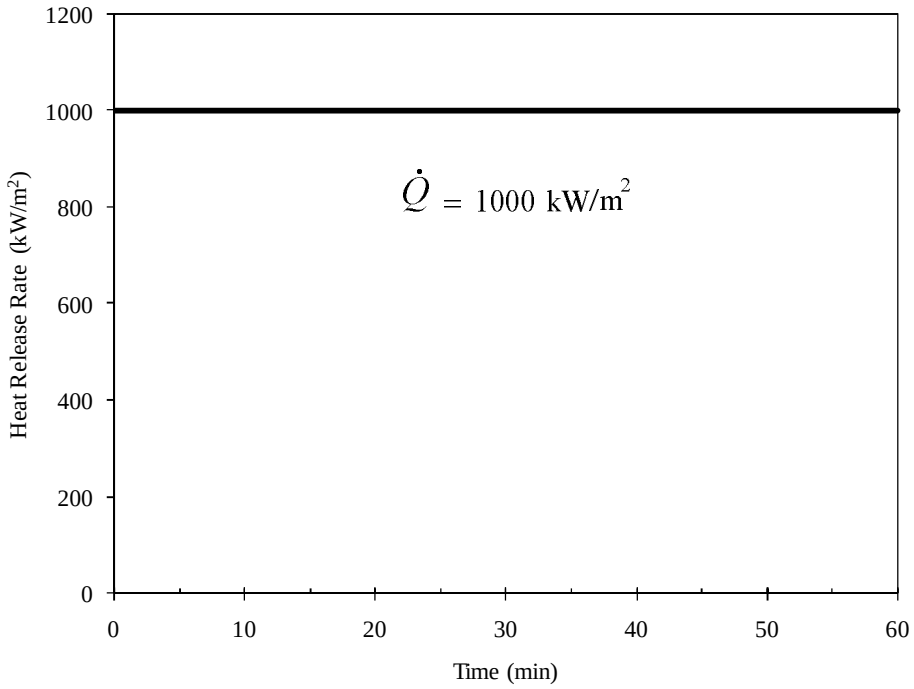


Figure 7-22. Fire load used for the simple beams

Curved Beam

Figure 7-23 shows the FE model mesh with fire loading area. The predicted temperature results from heat transfer analysis are shown in Figure 7-24. According to the different coat layer thickness, the temperatures are changed from 900 ^{o}C to 1200 ^{o}C at 60 min after fire ignited. The equivalent plastic strains are compared with coat layer thickness along with time in Figure 7-25. The von mises and equivalent plastic strain contours of the beam at 10,20, and 30 min are shown in Figure 7-26 and Figure 7-27 respectively with different coat layer thicknesses.

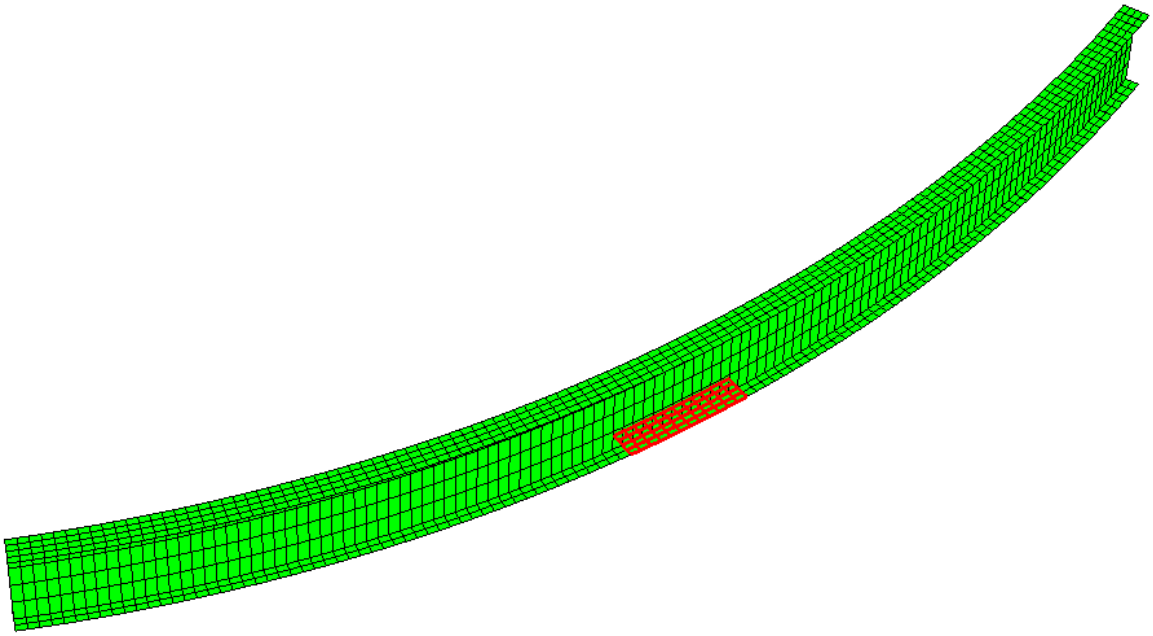


Figure 7-23. Curved beam model with Fire loading area (red)

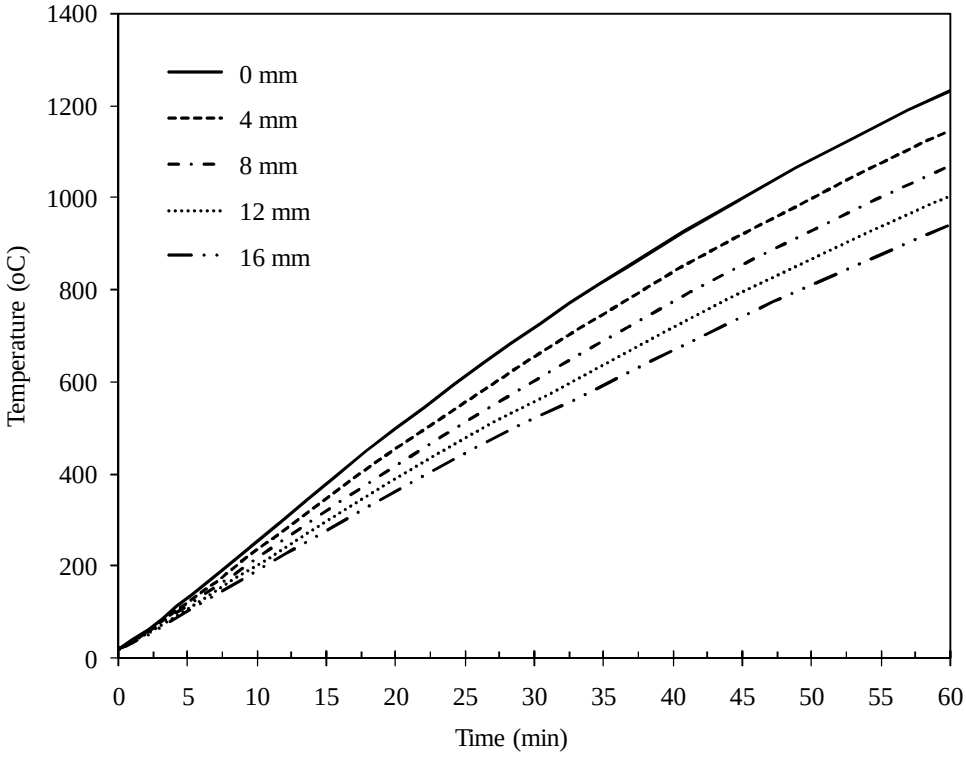


Figure 7-24. Bottom temperature profiles of a curved beam with different coat layer thickness

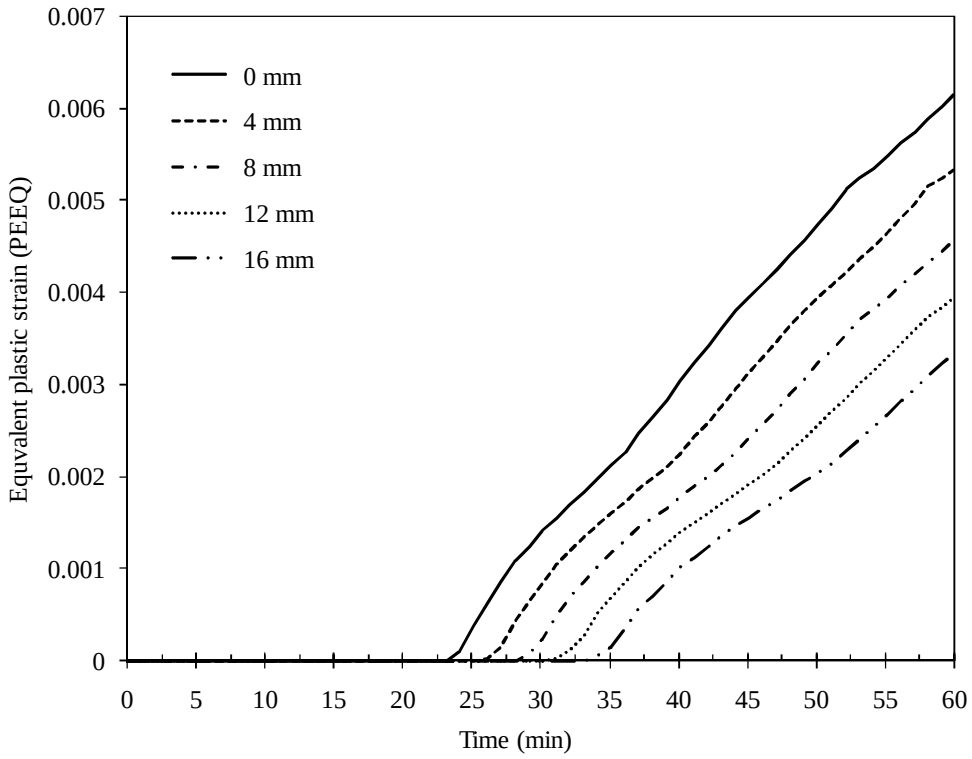


Figure 7-25. Equivalent plastic strain of a curved beam with different coat layer thickness

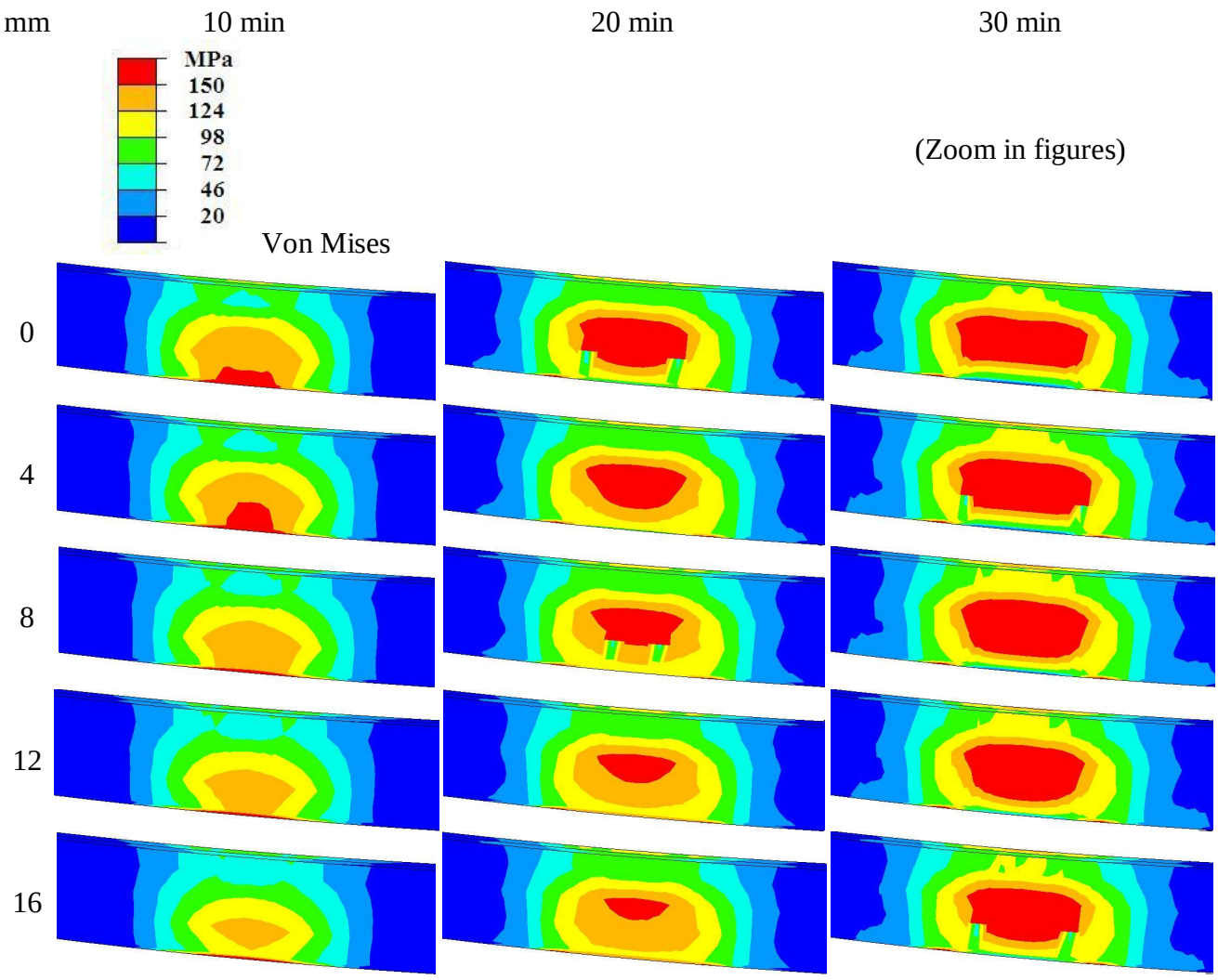


Figure 7-26. Von mises contours at 10, 20, and 30 min

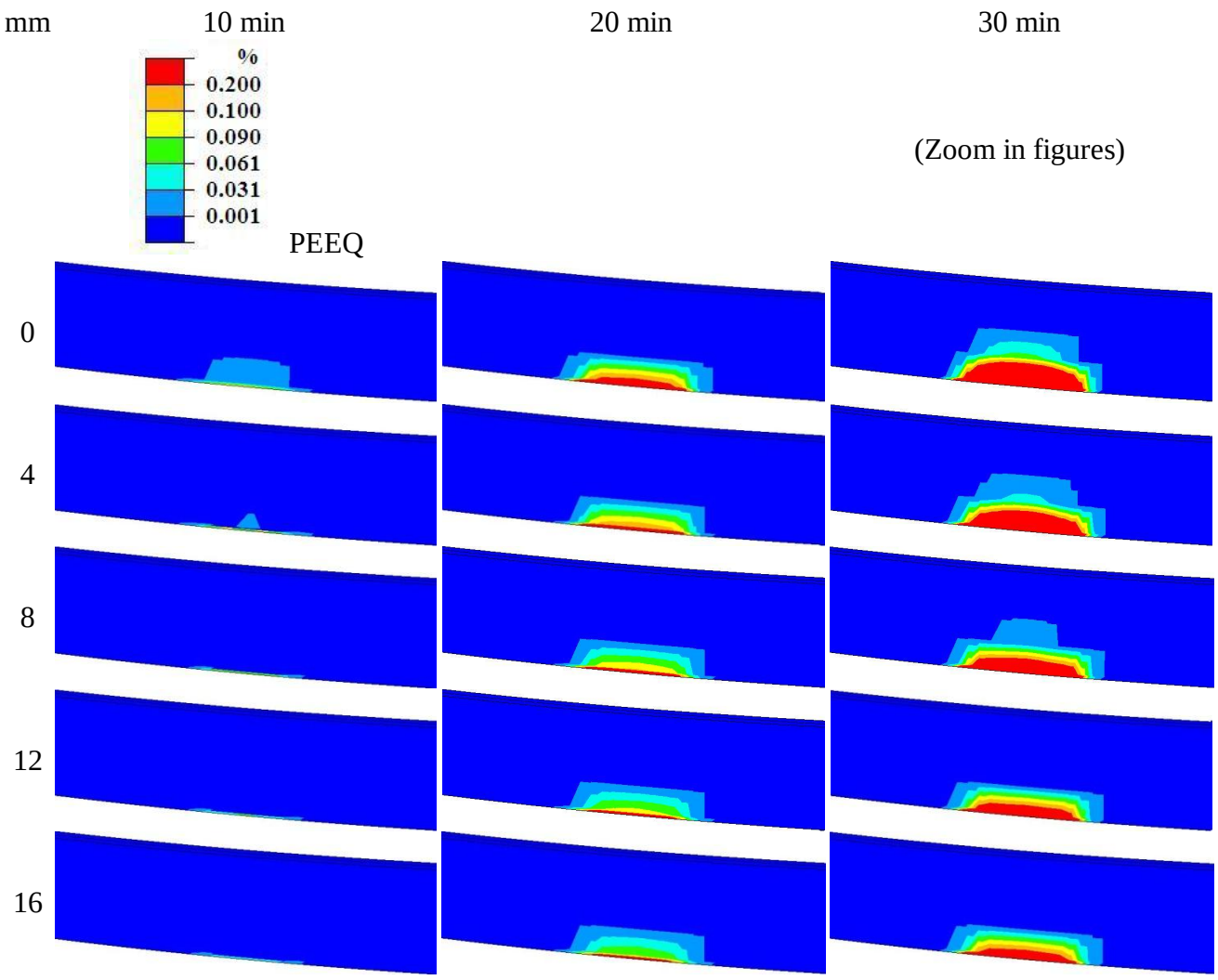


Figure 7-27. Equivalent plastic strain contours at 10, 20, and 30 min

Straight beam

Similar the previous curved beam, Figure 7-28 shows the FE model mesh with fire loading area. The temperature results from heat transfer analysis and the equivalent plastic strains are shown with coat layer thickness in Figure 7-29 and Figure 7-30 respectively. The von mises and equivalent plastic strain contours of the beam at 10,20, and 30 min are also shown in Figure 7-31 and Figure 7-32 respectively with different coat layer thicknesses.

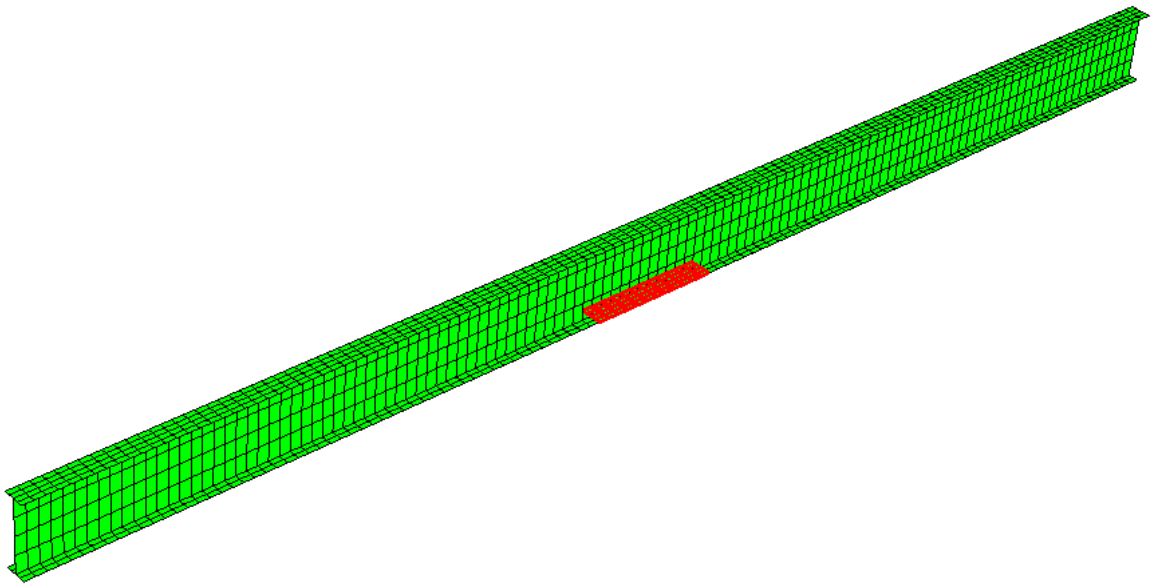


Figure 7-28. Straight beam with fire loading area (red)

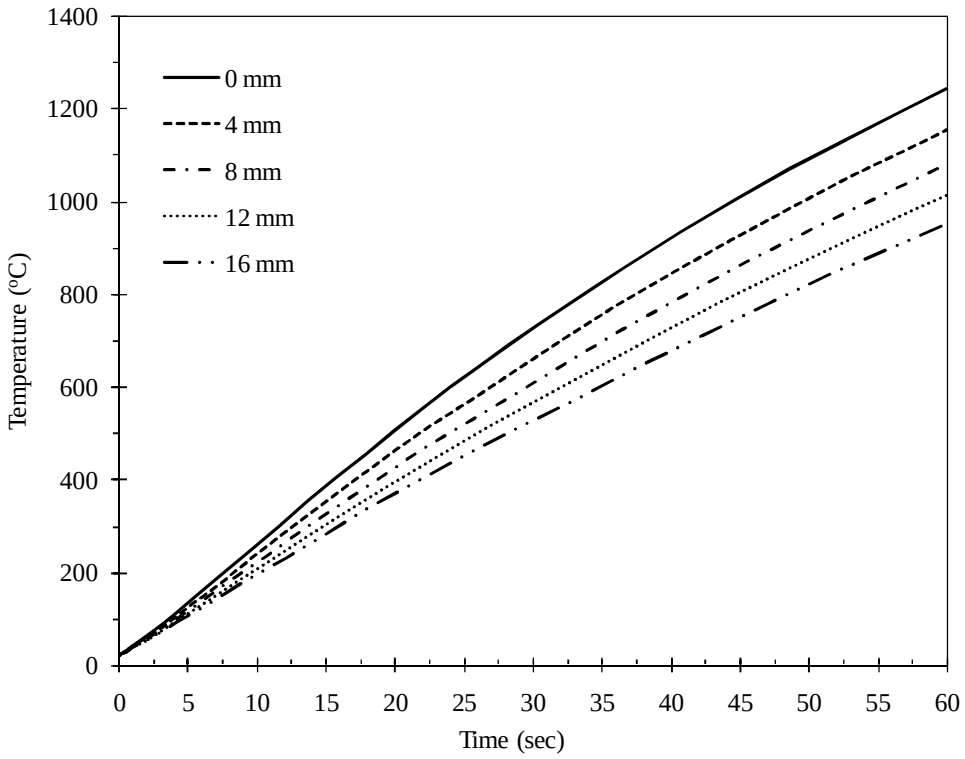


Figure 7-29. Bottom temperature profiles of a straight beam with different coat layer thickness

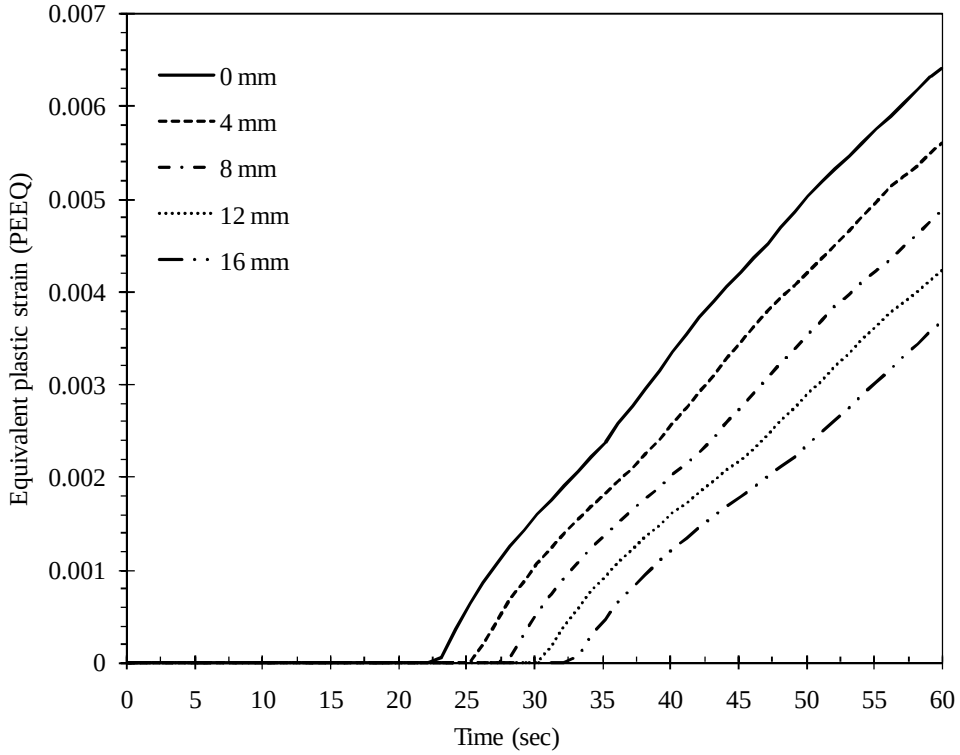


Figure 7-30. Equivalent plastic strain of a straight beam with different coat layer thickness

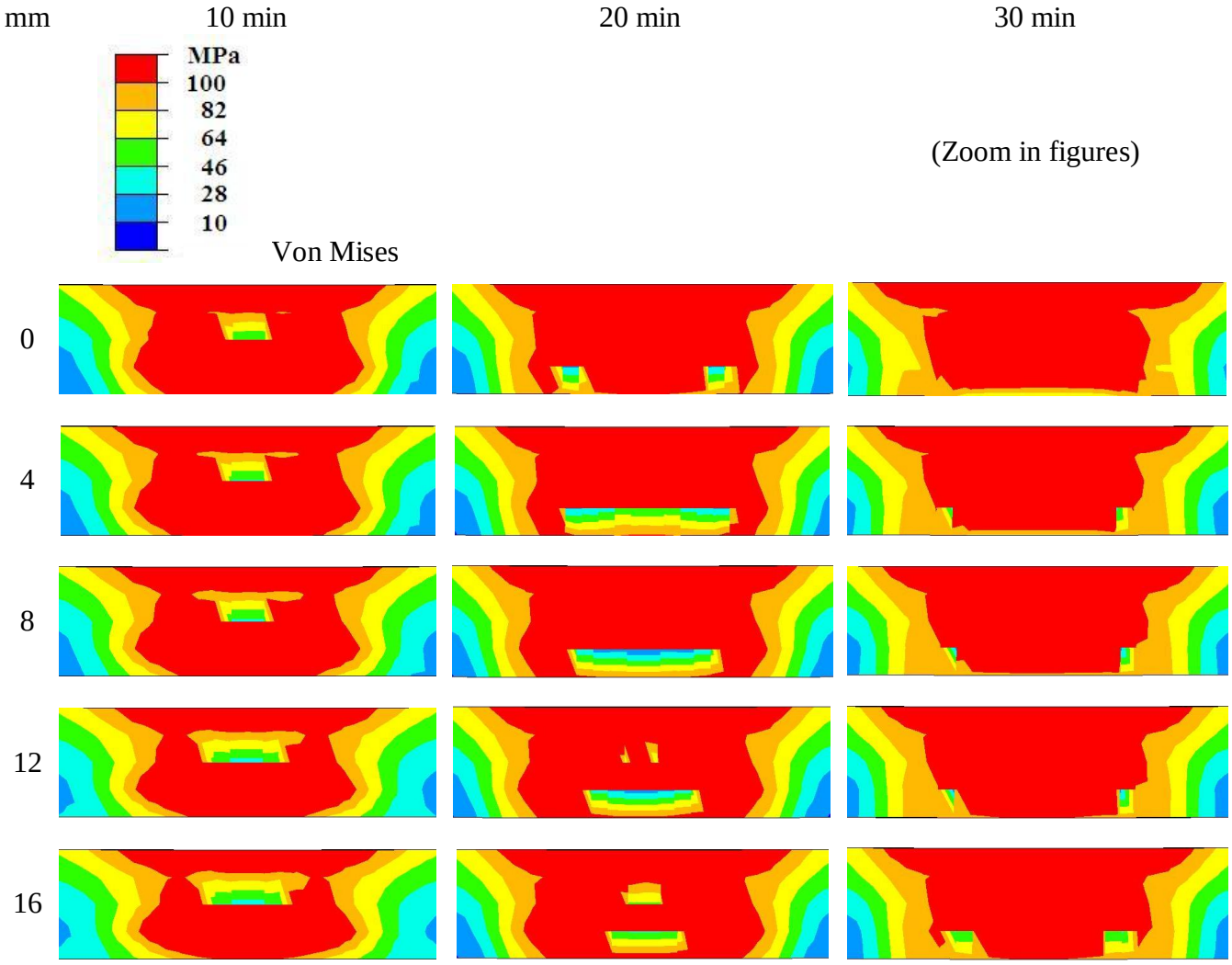


Figure 7-31. Von mises contours of a straight beam at 10, 20, and 30 min

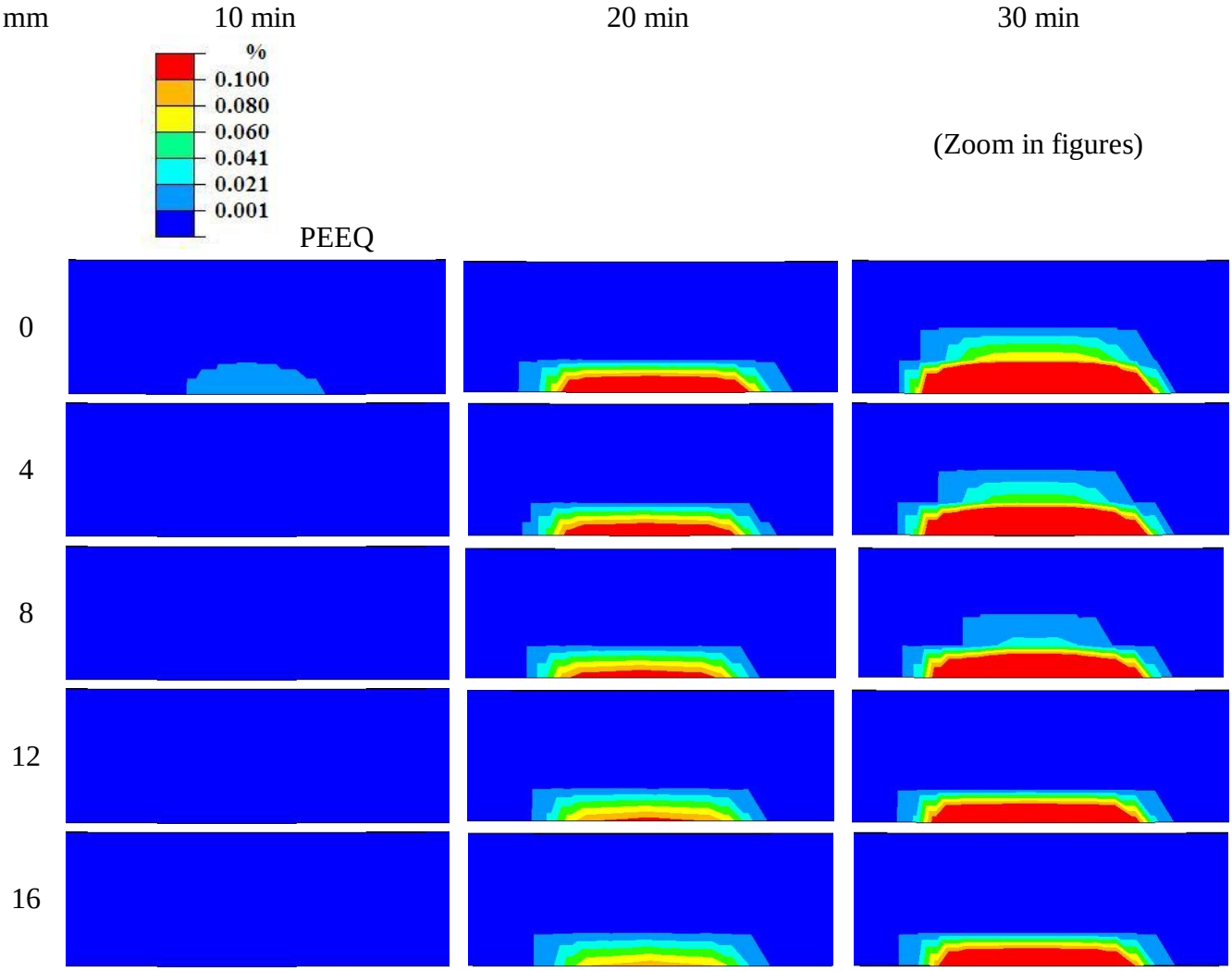


Figure 7-32. Equivalent plastic strain contours of a straight beam at 10, 20, and 30 min

CHAPTER 8

CONCLUSIONS AND FUTURE RECOMMENDATIONS

A new general Concurrent Fire-Structural (CFS) modeling framework is formulated, verified, and applied to study the behavior of civil structures under combined mechanical and thermal loadings due to fire. The proposed framework consists of three different modeling parts - fire simulation, heat transfer analysis, and thermal stress analysis. Fire is modeled and the time-dependent thermal results on the external surfaces are produced in the first fire simulation part. Heat distribution inside material is calculated using the fire simulation boundary conditions in an overall nonlinear transient heat transfer analysis. The last part of the proposed framework performs a thermomechanical stress analysis based on the entire temperature and heat distribution of the structures with and without additional mechanical loadings. The spatial-temporal temperature approximation scheme is applied to sequentially link the fire simulation and the thermomechanical finite element analysis. Thus the huge amount of thermal results from fire simulation is simplified using fourth-order polynomial approximations.

8.1 Conclusions

- A general concurrent fire – structural modeling framework is developed and can be applied to wide-scale structural systems, including steel-concrete buildings, bridges, sports stadiums, and tunnels. The few important cases that were studied strongly indicate the efficiency of the CFS framework under various fire scenarios.
- The new ST polynomial approximations provide a useful data reduction scheme that can "store" the spatial thermal history of the fire in a continuous manner. This is very important in order not to rely on a large amount of discrete numerical

data. The ST approximation can also be used as a mean to document fire histories for future scientific and forensic studies. It was found that the fourth order polynomials with spatial variables having fifth order temporal coefficients are very effective as they strike a good balance between accuracy and simplicity representing the surface thermal conditions of typical structural elements. The new spatial-temporal approximation, used in the subsequent structural analysis, allowed accurate and compact representations of the full-scale refined fire simulation.

- It is important to consider the dependency of the effective specific heat and conductivity on temperature in order to conduct realistic and accurate transient heat analysis in concrete materials. Towards that goal, this study collected wide range of concrete materials and efficiently represents their temperature dependencies. This can be useful in future studies that involve heat analysis in civil construction. Another important outcome of the study is the results of the simulation of high-strength concrete behavior as it shown to be very sensitive to fire due to the small aggregate composition. More attention should be directed to future constructions using this class of concrete.
- The proposed analysis framework has been applied to simulate tested normal strength and high strength concrete beams subjected to fire. Good correlation with the experimental data is shown by the proposed models. This makes the proposed CFS analysis framework well-suited for the study and design of new HSC construction under thermal loading including severe fire cases.
- The proposed analysis framework has been successfully applied to the third and fourth Cardington fire tests to examine its ability to predict the spatial-temporal distributions and nonlinear structural responses under fire loading during both the heating and cooling stages. Therefore, the CFS framework is able to accurately predict the deformation of steel structures under fire.

- The CFS framework was able to simulate the Oakland bridge collapse accident and the predicted the time-to-collapse was consisted with the reported time. The studied applications show that the proposed analysis framework can be used to study a wide range of civil structures under fire.
- The analysis framework can also be used in mitigation of fire damage in structures. A parametric study was carried to demonstrate that objective. A simply supported beam model was constructed to study the optimized coat layer needed to mitigate the damage and delay the time to reaching ultimate stress states.

8.2 Future recommendations

In the current CFS analysis, three-part simulations are concurrently carried out. Each simulation is separately but concurrently performed. However, it is possible to conduct fully coupled analysis where fire dynamics and thermomechanical structural analysis are coupled and conducted in a multi-physics analysis framework. This implies a unified formulation and coupling which in theory can be performed. In fact, the current separate formulations can be helpful in the development of this future coupled formulation. One of the benefits of a fully coupled formulation is that it will be possible to consider the effect of changing structural geometry, due to excessive deformations, on the fire progression and how these will alter the thermal distributions. For example, the changing of the structure geometry causes the changing of the fire flame, air flow, and surface temperatures in fire simulation. The proposed framework can be the basis to develop a fully coupled analysis approach.

Extensive structural fire design studies are needed to develop new code guidelines and reduce the probability of severe fire damage. Stochastic statistical analysis tools

can be used with the current modeling framework to develop these design guidelines. Moreover, coupled stochastic and CFS analysis can help determine the most important variables in the complicated structural responses that often are counterintuitive. For example, it is important to study the change in the location of fire source and many other factors that determine the fire scenario and how damage levels are changed in the structure.

The current analysis approach can be used to study the ventilation during fire as it can be important in order to increase the time-to-collapse of the structure. A new structural design such as changing ventilation systems, arrangements of windows, or doors is needed to consider the reduction of structural fire damage. Moreover, structures can be classified in damage resistance based on the potential fire source. For example: gas pipe, high voltage wire, or fire due to moving gas trucks. New analysis and design studies can be performed using the proposed framework.

The concrete tension and compression temperature-dependent properties are considered in the nonlinear finite element stress analysis part in the proposed framework. However, discrete concrete cracking which is one of the concrete main characteristics is not considered in our model under thermal mechanical loadings. It is recommended that more refined damage mechanics can be used selectively where thermal gradients occur and selective-adaptive mesh refinement can be applied for this purpose.

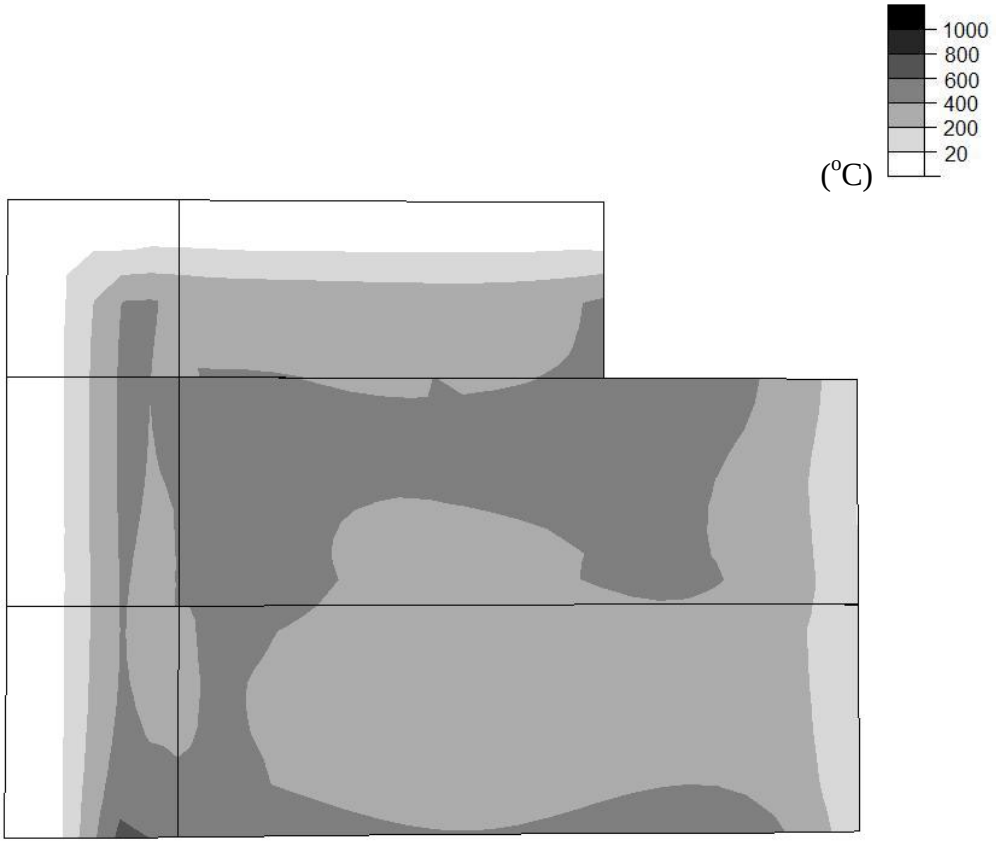
APPENDIX A

DETAILED FIGURES

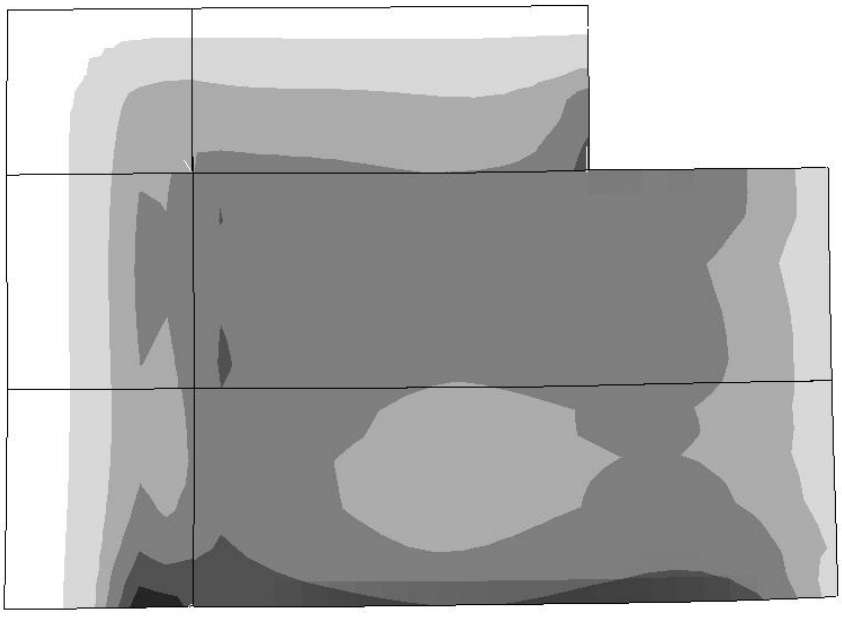
The detailed figures which are not shown in the main chapters are provided in this appendix A.

A.1 Detailed figures of the third Cardington fire test model

The detailed figures of showing in Figure 5-30 are given in this appendix A.1.

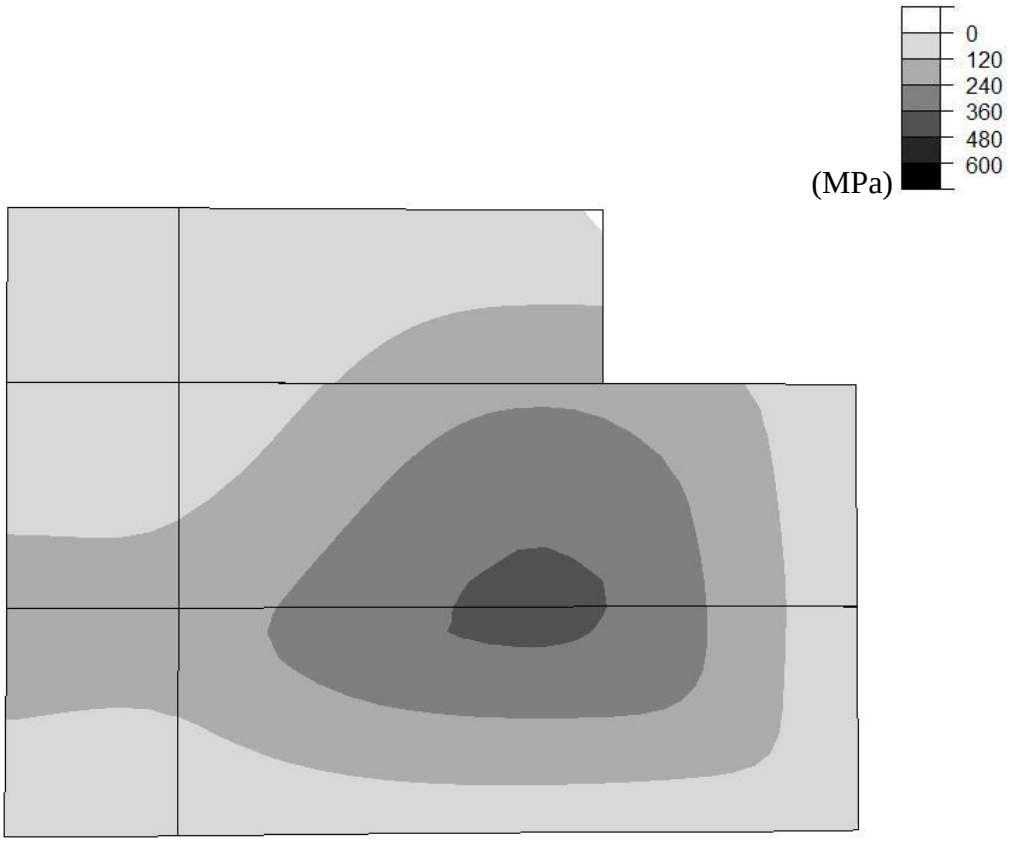


(a) at 60 min



(b) at 120 min

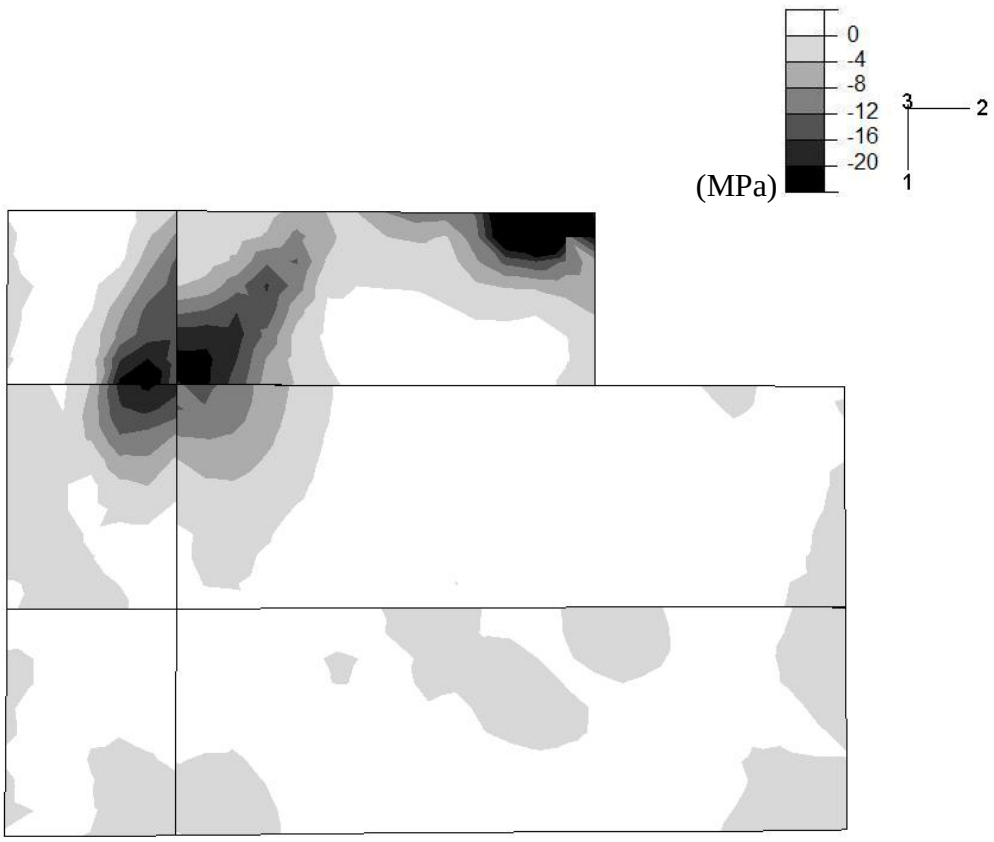
Figure A-1. Temperature at bottom of concrete slab for the third Cardington fire test



(a) at 60 min

(b) at 120 min

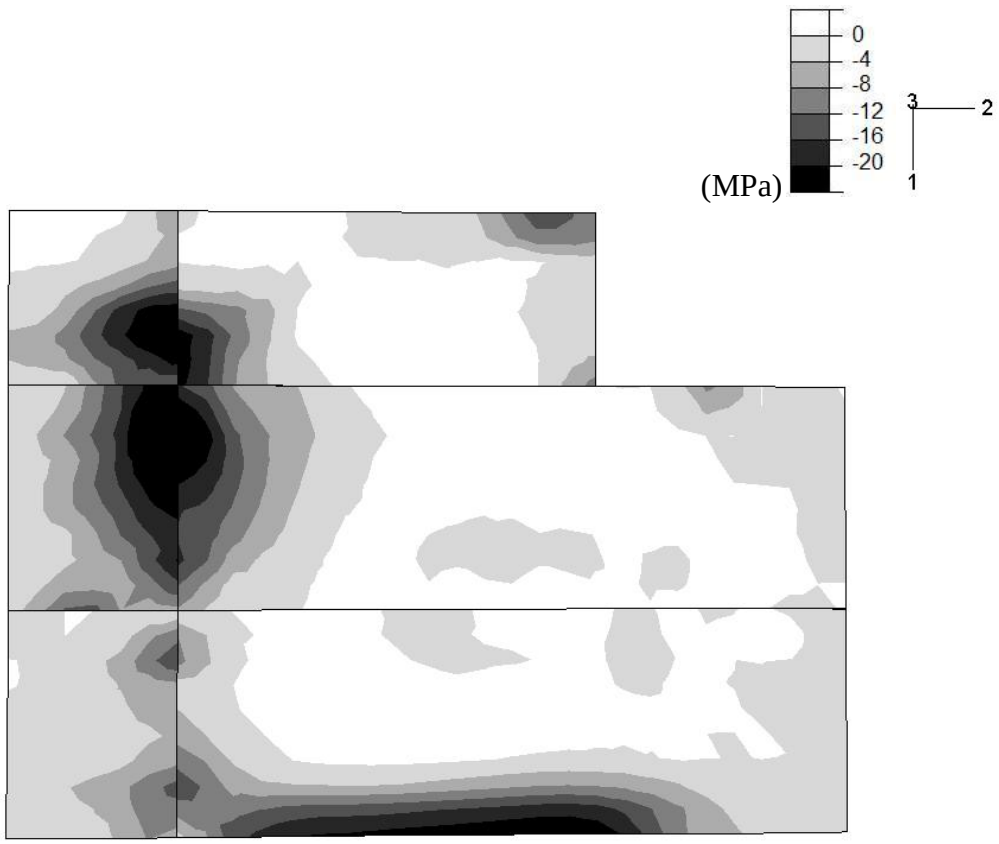
Figure A-2. vertical deflection contour



(a) at 60 min

(b) at 120 min

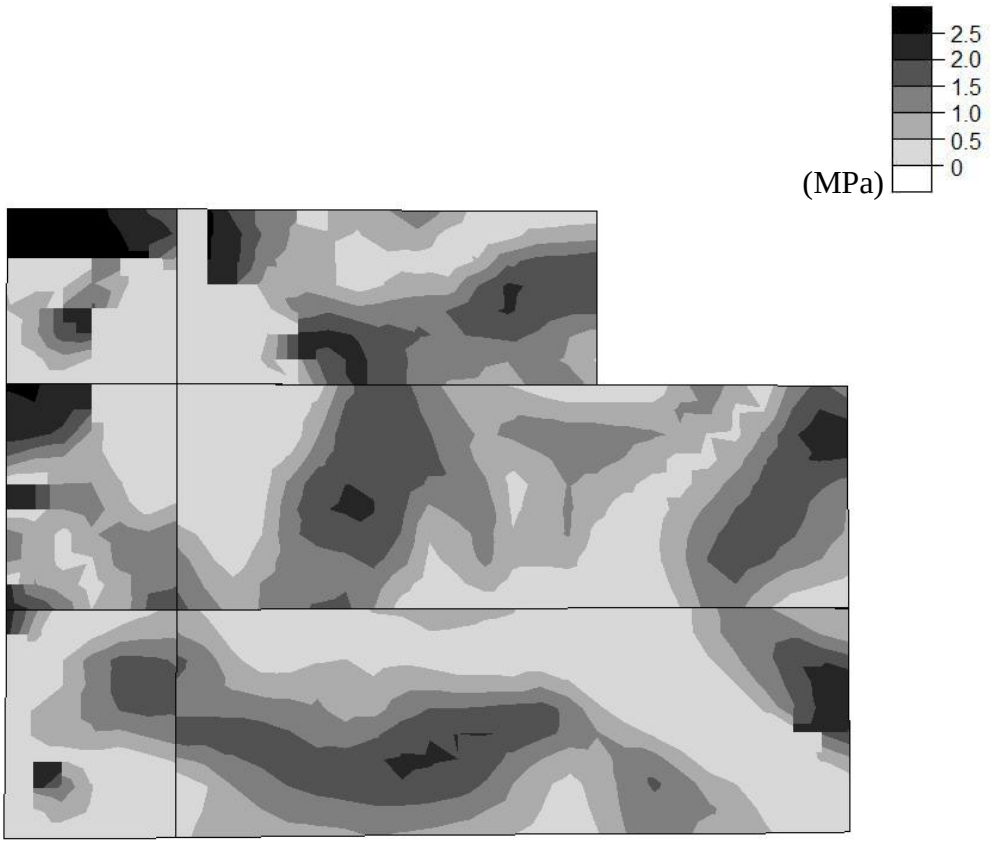
Figure A-3. 1-direction stress contour



(a) at 60 min

(b) at 120 min

Figure A-4. 2-direction stress contour



(a) at 60 min

(b) at 120 min

Figure A-5. Maximum principal stress contour

APPENDIX B

SOFTWARE DEVELOPED IN THIS STUDY

Several programs are developed in this study. These Fortran-based programs are used in part to link the general-purpose ABAQUS FE heat transfer and thermal stress analysis. The developed software are presented and described in this Appendix. In addition, stand alone example programs generating the Fire Dynamics Simulator (FDS) and ABAQUS bridge model are also presented.

These programs are needed to consider the nonlinear distributed flux and the thermal constitutive behavior of material in the heat transfer analysis. Another program linked with ABAQUS is also implemented to take into account the time-dependent temperature profile from Fire Dynamics Simulator (FDS) in the thermal stress analysis. The programs-subroutines called "DFLUX" and "UMATHT" are part of ABAQUS and allow the user to externally supply new behavior which is not developed as part of the FE code. We used those to develop our nonlinear transient heat transfer. Another subroutine that is developed, the UTEMP, and used in the thermal stress analysis and prescribe the fire boundary conditions using the ST polynomials.

A program "POST-FDS" is also developed to generate the ST polynomials and their coefficients based on least-square type error minimization. A coefficient matrix (5 by 6) is created for each column or beam element. "PreFDS" and "PreABAQUS" are developed to generate curved bridge model for FDS and ABAQUS.

B.1 DFLUX program linked with ABAQUS heat transfer analysis

The following "DFLUX" subroutine is used to apply the heat flux obtained from Fire Dynamics Simulator (FDS) as thermal loads in ABAQUS heat transfer analysis. The variation series called "A_COEF" contain the coefficients for time-dependent polynomial approximations described at chapter 2.4. The heat flux is calculated at a certain time and location in the subroutine and provided to the ABAQUS FE program based on these coefficients.

```
================================================================
DFLUX(FLUX,SOL,KSTEP,KINC,TIME,NOEL,NPT,COORDS,JLTYP,TEMP,PRESS,SNAME)
C
      INCLUDE 'ABA_PARAM.INC'
C
      DIMENSION FLUX(2), TIME(2), COORDS(3)
      CHARACTER*80 SNAME
C
C
C     TIME(1): Current value of step time.
C     TIME(2): Current value of total time.
C     NOEL   : Element number
C
      REAL*8 T,X,Y,Z
      REAL*8 A1(5),A_COEF1(6,5),A_COEF2(6,5),A_COEF3(6,5),A_COEF4(6,5)
      REAL*8 A_COEF5(6,5),A_COEF6(6,5),A_COEF7(6,5),A_COEF8(6,5)
      REAL*8 A_COEF9(6,5),A_COEF10(6,5),A_COEF11(6,5),A_COEF12(6,5)
      REAL*8 A_COEF13(6,5),A_COEF14(6,5),A_COEF15(6,5),A_COEF16(6,5)
      REAL*8 A_COEF17(6,5),A_COEF18(6,5)
      REAL*8 QTEMP

      DATA A_COEF1 /0.407472508283467E+04,   -0.147529546730421E+02,
     &    0.449878990365782E-02,   -0.269134700458179E-06,
     &   -0.369509043863781E-10,    0.358374861994802E-14,
     &   -0.413754778161746E+03,    0.149649714147115E+01,
     &   -0.454744402951061E-03,    0.271522075036961E-07,
     &    0.373510488555242E-11,   -0.362061171033273E-15,
     &    0.157223929996314E+02,   -0.568030883513273E-01,
     &    0.171914768466997E-04,   -0.102054938894967E-08,
     &   -0.141959809921585E-12,    0.137209570436392E-16,
     &   -0.264983278241452E+00,    0.956189230228686E-03,
     &   -0.288008615837198E-06,    0.169085956883095E-10,
     &    0.240779257383371E-14,   -0.231331368194832E-18,
     &    0.167120751952290E-02,   -0.602247373723098E-05,
     &    0.180381105964704E-08,   -0.104105570683914E-12,
     &   -0.153806550736375E-16,    0.146420045487082E-20 /

      DATA A_COEF2 /0.394529250097868E+04,   -0.153799688528902E+02,
```

```
&    0.668023485044511E-02,   -0.106302606889676E-05,
&    0.681319647971784E-10,   -0.121451520930361E-14,
&   -0.400674806010473E+03,    0.155841161812452E+01,
&   -0.672215457354963E-03,    0.106306614863059E-06,
&   -0.673981847095410E-11,    0.116114552335247E-15,
&    0.152262932738709E+02,   -0.590822634910335E-01,
&    0.252960903013126E-04,   -0.397139804612782E-08,
&    0.248482584085170E-12,   -0.409840792041642E-17,
&   -0.256617213433024E+00,    0.993263795285165E-03,
&   -0.421833920845560E-06,    0.656583808197056E-10,
&   -0.404198476869323E-14,    0.629697829131730E-19,
&    0.161830780968467E-02,   -0.624737295811017E-05,
&    0.262996346040120E-08,   -0.405257860021541E-12,
&    0.244617579577031E-16,   -0.353479068256877E-21 /

 DATA A_COEF3 /0.376863428733381E+04,   -0.152133803597062E+02,
&    0.749702777124561E-02,   -0.140822851801402E-05,
&    0.116957321916304E-09,   -0.354056517016136E-14,
&   -0.382971818130056E+03,    0.154148126415157E+01,
&   -0.753780230096202E-03,    0.140768032776058E-06,
&   -0.116112825433364E-10,    0.348069238574945E-15,
&    0.145614193935453E+02,   -0.584341866107607E-01,
&    0.283409169864763E-04,   -0.525801398360327E-08,
&    0.430284092441932E-12,   -0.127510619731982E-16,
&   -0.245527287506739E+00,    0.982200459948124E-03,
&   -0.472194571451783E-06,    0.869501259858773E-10,
&   -0.704977156195376E-14,    0.206071405752771E-18,
&    0.154900712147782E-02,   -0.617639153577178E-05,
&    0.294137015864328E-08,   -0.537036395769786E-12,
&    0.430752724023488E-16,   -0.123881284360214E-20 /

 DATA A_COEF4 /0.378172146835574E+04,   -0.155481857145178E+02,
&    0.829471372118373E-02,   -0.173252892527354E-05,
&    0.164726418405491E-09,   -0.592449722270051E-14,
&   -0.383727435293826E+03,    0.157263493128380E+01,
&   -0.832167940100936E-03,    0.172772193352986E-06,
&   -0.163289761427783E-10,    0.583519416853780E-15,
&    0.145677686553026E+02,   -0.595090635348683E-01,
&    0.312215502228531E-04,   -0.643979156755917E-08,
&    0.604654948207671E-12,   -0.214549962927193E-16,
&   -0.245251540259190E+00,    0.998480013805203E-03,
&   -0.519119092362124E-06,    0.106304212949845E-09,
&   -0.990876658311415E-14,    0.348821533762646E-18,
&    0.154484455327120E-02,   -0.626758696191452E-05,
&    0.322729624762113E-08,   -0.655659383088831E-12,
&    0.606218791070949E-16,   -0.211525592284514E-20 /

 DATA A_COEF5 /0.376513072362728E+04,   -0.149674281183597E+02,
&    0.730317875848163E-02,   -0.137125660874338E-05,
&    0.114602653471238E-09,   -0.351440300789807E-14,
&   -0.381719914971167E+03,    0.151347319606936E+01,
&   -0.732879541769704E-03,    0.136697646157085E-06,
&   -0.113294606247934E-10,    0.343272714230517E-15,
&    0.144799198332212E+02,   -0.572570934948846E-01,
&    0.275043500013970E-04,   -0.509290062879770E-08,
&    0.418189179946775E-12,   -0.124996444010007E-16,
```

```
&  -0.243587574148442E+00,    0.960517057921079E-03,
&  -0.457454991173738E-06,    0.840199577007585E-10,
&  -0.682682183438592E-14,    0.200884607522295E-18,
&   0.153326516798441E-02,   -0.602843914625012E-05,
&   0.284485363602430E-08,   -0.517807003067492E-12,
&   0.415751562594167E-16,   -0.120145077689420E-20 /

 DATA A_COEF6 /0.344873454527766E+04,   -0.129332540612072E+02,
&   0.497760568435490E-02,   -0.616094023306768E-06,
&   0.165370623341281E-10,    0.984986949527774E-15,
&  -0.349817991483462E+03,    0.130915382534405E+01,
&  -0.500056091630396E-03,    0.611734024508301E-07,
&  -0.152750116799446E-11,   -0.106277564605187E-15,
&   0.132774427152535E+02,   -0.495840568277330E-01,
&   0.187888203556962E-04,   -0.226851413004146E-08,
&   0.518216267829779E-13,    0.429702486764357E-17,
&  -0.223505163326216E+00,    0.832827371792536E-03,
&  -0.312871084004305E-06,    0.372097271761700E-10,
&  -0.757849032783990E-15,   -0.772621612890050E-19,
&   0.140787357556418E-02,   -0.523392099078428E-05,
&   0.194797982314293E-08,   -0.227698062612258E-12,
&   0.398004935673886E-17,    0.521028503875334E-21 /

 DATA A_COEF7 /0.273591324867786E+04,   -0.921916294933300E+01,
&   0.154251076460786E-02,    0.360063651191356E-06,
&  -0.972708340950765E-10,    0.574342386024553E-14,
&  -0.278226370337831E+03,    0.935760006384594E+00,
&  -0.154438399845041E-03,   -0.371544970453203E-07,
&   0.995028591113188E-11,   -0.586765453099334E-15,
&   0.105887793867017E+02,   -0.355480421595797E-01,
&   0.578814601319704E-05,    0.143422742427278E-08,
&  -0.380906924650454E-12,    0.224335762633820E-16,
&  -0.178755520799307E+00,    0.599016871570913E-03,
&  -0.961876814954064E-07,   -0.245682195472588E-10,
&   0.647009651663736E-14,   -0.380545393305925E-18,
&   0.112936247253874E-02,   -0.377763496551908E-05,
&   0.597792782674249E-09,    0.157624066283483E-12,
&  -0.411505695822526E-16,    0.241676050667940E-20 /

 DATA A_COEF8 /0.263375375722384E+04,   -0.852451132968361E+01,
&   0.776325619521234E-03,    0.584984235712659E-06,
&  -0.123986416459144E-09,    0.688288880768502E-14,
&  -0.267817860013995E+03,    0.865134440159338E+00,
&  -0.765970747118178E-04,   -0.600572087164521E-07,
&   0.126773560665587E-10,   -0.703347043553247E-15,
&   0.101926804765574E+02,   -0.328647080017328E-01,
&   0.283210039254793E-05,    0.230591119639106E-08,
&  -0.484951233316195E-12,    0.268914133008517E-16,
&  -0.172082411948683E+00,    0.553861658518270E-03,
&  -0.464550667689413E-07,   -0.392660130745392E-10,
&   0.822858276726012E-14,   -0.456053410868778E-18,
&   0.108736638094978E-02,   -0.349364996912261E-05,
&   0.285019335414878E-09,    0.250263172835488E-12,
&  -0.522598295292416E-16,    0.289479854215161E-20 /

 DATA A_COEF9 /0.266526803425304E+04,   -0.800851883403357E+01,
```

```
&  -0.378861663496988E-03,    0.101051360587654E-05,
&  -0.182979004957100E-09,    0.971916872528940E-14,
&  -0.270648871500933E+03,    0.811573387594649E+00,
&   0.411561393189042E-04,   -0.103319785437208E-06,
&   0.186668648107935E-10,   -0.991038969944955E-15,
&   0.102874093841992E+02,   -0.307909855541446E-01,
&  -0.165405362803281E-05,    0.395029983427801E-08,
&  -0.712337622826923E-12,    0.378041850551942E-16,
&  -0.173482123063053E+00,    0.518362364614250E-03,
&   0.292548006717685E-07,   -0.669606251269484E-10,
&   0.120542185105138E-13,   -0.639519446554303E-18,
&   0.109507346828353E-02,   -0.326689427236973E-05,
&  -0.192543751702318E-09,    0.424644179564266E-12,
&  -0.763266299654905E-16,    0.404824136520949E-20 /

 DATA A_COEF10 /0.125056272248033E+04,   -0.246853062028094E+01,
&  -0.331239259018079E-02,    0.156929366083999E-05,
&  -0.223677103666611E-09,    0.105016659454611E-13,
&  -0.128859121940652E+03,    0.259521127760365E+00,
&   0.328034982822079E-03,   -0.156644886148866E-06,
&   0.223728391390084E-10,   -0.105159161863848E-14,
&   0.496845283006724E+01,   -0.101996772177149E-01,
&  -0.121524889703769E-04,    0.585198101234932E-08,
&  -0.837632168640410E-12,    0.394185450684467E-16,
&  -0.849699158637378E-01,    0.177626440803047E-03,
&   0.199659267583060E-06,   -0.969970428968651E-10,
&   0.139157407911440E-13,   -0.655698135858947E-18,
&   0.543786462766482E-03,   -0.115635620681056E-05,
&  -0.122770119953438E-08,    0.601937558065934E-12,
&  -0.865642177726035E-16,    0.408423002563611E-20 /

 DATA A_COEF11 /0.142920120156516E+04,   -0.317108056595998E+01,
&  -0.284600463526230E-02,    0.146521540607370E-05,
&  -0.214429819105171E-09,    0.102326578120144E-13,
&  -0.146893259024328E+03,    0.330293298878075E+00,
&   0.281473246004194E-03,   -0.146274914089657E-06,
&   0.214505761978617E-10,   -0.102466010736892E-14,
&   0.564853791231440E+01,   -0.128632712156431E-01,
&  -0.104149259375491E-04,    0.546568229027355E-08,
&  -0.803237139831846E-12,    0.384101646931904E-16,
&  -0.963252732313649E-01,    0.222021611653633E-03,
&   0.170928104555604E-06,   -0.906187865922508E-10,
&   0.133470350810376E-13,   -0.638953099762996E-18,
&   0.614625015497028E-03,   -0.143288231235002E-05,
&  -0.105005698758765E-08,    0.562544981978072E-12,
&  -0.830457486838775E-16,    0.398014076480588E-20 /

 DATA A_COEF12 /0.158579576457327E+04,   -0.366283069367091E+01,
&  -0.268510710673864E-02,    0.145747174046479E-05,
&  -0.216856074952677E-09,    0.104585244300611E-13,
&  -0.162597647686285E+03,    0.379498586410620E+00,
&   0.265684610748940E-03,   -0.145597225732626E-06,
&   0.217036175179631E-10,   -0.104761672299620E-14,
&   0.623713732214219E+01,   -0.147029571149551E-01,
&  -0.983675771356728E-05,    0.544419865612571E-08,
&  -0.813115880691537E-12,    0.392835665895368E-16,
```

```
&   -0.106097099408252E+00,    0.252484860270491E-03,
&    0.161560183937906E-06,   -0.903290919155273E-10,
&    0.135178993879366E-13,   -0.653685296434106E-18,
&    0.675261328588583E-03,   -0.162141147314596E-05,
&   -0.993369978626771E-09,    0.561168026626425E-12,
&   -0.841495227180444E-16,    0.407305676427748E-20 /

 DATA A_COEF13 /0.165274611989746E+04,   -0.368735710377996E+01,
&   -0.303523242425952E-02,    0.159251162929537E-05,
&   -0.235034899554541E-09,    0.112956619526533E-13,
&   -0.169246864296147E+03,    0.381332200357804E+00,
&    0.301794685846679E-03,   -0.159475364929436E-06,
&    0.235702363547528E-10,   -0.113355361180967E-14,
&    0.648370083238478E+01,   -0.147476540915168E-01,
&   -0.112278477365536E-04,    0.597686940350180E-08,
&   -0.884684136544918E-12,    0.425770666661260E-16,
&   -0.110143881357580E+00,    0.252824444722466E-03,
&    0.185279204876679E-06,   -0.993773655492420E-10,
&    0.147321084759177E-13,   -0.709528778096713E-18,
&    0.700071275881897E-03,   -0.162102603885384E-05,
&   -0.114438757824742E-08,    0.618559776752990E-12,
&   -0.918405717571382E-16,    0.442652046319355E-20 /

 DATA A_COEF14 /0.168872928933625E+04,   -0.355086611788238E+01,
&   -0.354407513241251E-02,    0.177611630395843E-05,
&   -0.259402353510450E-09,    0.124165572858239E-13,
&   -0.172875018083010E+03,    0.367532337562864E+00,
&    0.353137477854815E-03,   -0.178024599307967E-06,
&    0.260347667632552E-10,   -0.124702707368699E-14,
&    0.662049870373903E+01,   -0.142261416256382E-01,
&   -0.131648859990931E-04,    0.667750073814473E-08,
&   -0.977867538958959E-12,    0.468711196952209E-16,
&   -0.112429855240208E+00,    0.244093985664828E-03,
&    0.217663520451177E-06,   -0.111103458599238E-09,
&    0.162931166780803E-13,   -0.781518208300465E-18,
&    0.714356276125150E-03,   -0.156639944209674E-05,
&   -0.134682043066257E-08,    0.691932165411130E-12,
&   -0.101616373551722E-15,    0.487766874073348E-20 /

 DATA A_COEF15 /0.174443740754249E+04,   -0.333237082211963E+01,
&   -0.430612295564847E-02,    0.208073437729856E-05,
&   -0.303530731429324E-09,    0.145965360442473E-13,
&   -0.178412320530100E+03,    0.345307178644362E+00,
&    0.429927268232322E-03,   -0.208695566502421E-06,
&    0.304761934565916E-10,   -0.146638026509418E-14,
&    0.682649771824344E+01,   -0.133811224547955E-01,
&   -0.160591490177345E-04,    0.783263447343865E-08,
&   -0.114507956130977E-11,    0.551273218208064E-16,
&   -0.115829344616628E+00,    0.229861420324551E-03,
&    0.266020616221689E-06,   -0.130389618407158E-09,
&    0.190839209857335E-13,   -0.919282989682206E-18,
&    0.735353309616271E-03,   -0.147679955537139E-05,
&   -0.164900618789775E-08,    0.812372572771930E-12,
&   -0.119038966485270E-15,    0.573751845572837E-20 /

 DATA A_COEF16 /0.204640189253143E+04,   -0.438201057780861E+01,
```

```
&   -0.395629006828591E-02,    0.208677021295949E-05,
&   -0.314113766668498E-09,    0.154047354782664E-13,
&   -0.208271433797927E+03,    0.448183821648286E+00,
&    0.397463613920013E-03,   -0.210024741352196E-06,
&    0.316165883191444E-10,   -0.155047977366477E-14,
&    0.793155968771589E+01,   -0.171555158855874E-01,
&   -0.149348109389023E-04,    0.790827365338026E-08,
&   -0.119066393319379E-11,    0.583897184512339E-16,
&   -0.133971314157776E+00,    0.291300264675665E-03,
&    0.248796802468029E-06,   -0.132053360020460E-09,
&    0.198859842638751E-13,   -0.975215148786899E-18,
&    0.846831948957072E-03,   -0.185120032560895E-05,
&   -0.155054145101155E-08,    0.825106278572449E-12,
&   -0.124285580692802E-15,    0.609520619992412E-20 /

 DATA A_COEF17 /0.230107931596600E+04,   -0.574532017540423E+01,
&   -0.264363808750945E-02,    0.171875114143274E-05,
&   -0.271995065640486E-09,    0.136742221868079E-13,
&   -0.233302242009129E+03,    0.582600151952136E+00,
&    0.267771092798880E-03,   -0.173604567429633E-06,
&    0.274413675886442E-10,   -0.137863199900867E-14,
&    0.885256694825011E+01,   -0.221171350648106E-01,
&   -0.101374982259175E-04,    0.655892696902723E-08,
&   -0.103571731354163E-11,    0.520012516502227E-16,
&   -0.149008557735250E+00,    0.372566454849538E-03,
&    0.170057031368906E-06,   -0.109871078811348E-09,
&    0.173346329801076E-13,   -0.869845306856667E-18,
&    0.938754050058332E-03,   -0.234954582622646E-05,
&   -0.106668475096494E-08,    0.688585114229123E-12,
&   -0.108558413843467E-15,    0.544461801346145E-20 /

 DATA A_COEF18 /0.217625439497083E+04,   -0.368494172944702E+01,
&   -0.608572352837200E-02,    0.265272556897785E-05,
&   -0.377304200006894E-09,    0.181065240513485E-13,
&   -0.219232166639878E+03,    0.370561053623987E+00,
&    0.615490878931091E-03,   -0.267885859730381E-06,
&    0.380575935738607E-10,   -0.182444317329600E-14,
&    0.826778944390026E+01,   -0.139553412181925E-01,
&   -0.232922752252396E-04,    0.101238499867016E-07,
&   -0.143667152835577E-11,    0.688038194446143E-16,
&   -0.138355240232272E+00,    0.233290819705134E-03,
&    0.390953700316055E-06,   -0.169714284217143E-09,
&    0.240589209263940E-13,   -0.115110745363177E-17,
&    0.866829018608684E-03,   -0.146059824016227E-05,
&   -0.245576336043467E-08,    0.106485134414770E-11,
&   -0.150805507696091E-15,    0.720874282999644E-20 /

 T = TIME(1)
 X = COORDS(1)
 Y = COORDS(2)
 Z = COORDS(3)

 DO I=1,5
   A1(I)=0.0
 ENDDO
 QTEMP = 0.0
```

```
SELECT CASE(NOEL)
    CASE(8272001:8272024)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF1(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272101:8272124)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF2(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272201:8272224)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF3(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272301:8272324)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF4(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272401:8272424)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF5(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272501:8272524)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF6(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272601:8272624)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF7(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272701:8272724)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF8(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
    CASE(8272801:8272824)
      DO I=1,5
      DO J=1,6
        A1(I)=A1(I)+A_COEF9(J,I)*T**FLOAT(J-1)
      ENDDO
      ENDDO
```

```
CASE(8322001:8322024)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF10(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322101:8322124)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF11(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322201:8322224)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF12(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322301:8322324)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF13(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322401:8322424)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF14(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322501:8322524)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF15(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322601:8322624)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF16(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322701:8322724)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF17(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
CASE(8322801:8322824)
  DO I=1,5
  DO J=1,6
    A1(I)=A1(I)+A_COEF18(J,I)*T**FLOAT(J-1)
  ENDDO
  ENDDO
```

```
      END SELECT

      DO I=1,5
        QTEMP=QTEMP+A1(I)*Y**FLOAT(I-1)
      ENDDO


FLUX(1) = QTEMP

RETURN
END
```

B.2 UMATHT program linked with ABAQUS heat transfer analysis

The program UMATHT is used in ABAQUS heat transfer analysis to provide the temperature – dependent mechanical and thermal properties of concrete. The ABAQUS FE calculates the temperatures inside of the concrete using the subroutine.

```
==============================================================
UMATHT(U,DUDT,DUDG,FLUX,DFDT,DFDG,STATEV,TEMP,DTEMP,DTEMDX,TIME,DTIME,PR
EDEF,DPRED,CMNAME,NTGRD,NSTATV,
PROPS,NPROPS,COORDS,PNEWDT,NOEL,NPT,LAYER,KSPT,KSTEP,KINC)
C
      INCLUDE 'ABA_PARAM.INC'
C
      CHARACTER*80 CMNAME
C
      DIMENSION DUDG(NTGRD),FLUX(NTGRD),DFDT(NTGRD),
     $     DFDG(NTGRD,NTGRD),STATEV(NSTATV),DTEMDX(NTGRD),TIME(2),
     $     PREDEF(1),DPRED(1),PROPS(NPROPS),COORDS(3)
C
C
      REAL*8 ROI
      REAL*8 CONDX(NTGRD)
      REAL*8 SPECHTGX,SPECHTGY,SPECHTGZ
      REAL*8 DRODT , FX , CEFF
      REAL*8 TEMP1(NTGRD),RON(10000)
      REAL*8 DMDT(NTGRD),DTEMDY(NTGRD),DTEMDZ(NTGRD)
C
C
C
      CONDX(1) = PROPS(1)
      CONDX(2) = PROPS(2)
      CONDX(3) = PROPS(3)
      SPECHT = PROPS(4)
      ROI = PROPS(5)

C
C
C input DU and DUDT
C
      IF (TEMP.LT.103) THEN
       DUDT = 1014.337 * (TEMP-20) + 1720000
      ENDIF
      IF ((TEMP.GE.103).AND.(TEMP.LT.165)) THEN
       DUDT = 10481.77 * (TEMP-103) + 1800000
      ENDIF
      IF ((TEMP.GE.165).AND.(TEMP.LT.440)) THEN
       DUDT = 827.4909 * (TEMP-165) + 2450000
      ENDIF
      IF ((TEMP.GE.440).AND.(TEMP.LT.480)) THEN
       DUDT = 17847 * (TEMP-440) + 2680000
```

```
      ENDIF
      IF ((TEMP.GE.480).AND.(TEMP.LT.506)) THEN
       DUDT = 14910 * (TEMP-480) + 3390000
      ENDIF
      IF ((TEMP.GE.506).AND.(TEMP.LT.523)) THEN
       DUDT = -10167.1 * (TEMP-506) + 3780000
      ENDIF
      IF ((TEMP.GE.523).AND.(TEMP.LT.576)) THEN
       DUDT = -16552.5 * (TEMP-523) + 3610000
      ENDIF
      IF ((TEMP.GE.576).AND.(TEMP.LT.755)) THEN
       DUDT = -166.257 * (TEMP-576) + 2730000
      ENDIF
      IF ((TEMP.GE.755).AND.(TEMP.LT.870)) THEN
       DUDT = -2386.09 * (TEMP-755) + 2700000
      ENDIF
      IF ((TEMP.GE.870).AND.(TEMP.LT.1000)) THEN
       DUDT = 40.61538 * (TEMP-870) + 2420000
      ENDIF
      IF (TEMP.GE.1000) THEN
       DUDT = 2430000
      ENDIF
    DU = DUDT*DTEMP
       U = U+DU

      DO i=1,NTGRD
          FLUX(i) = (-CONDX(i)*DTEMDX(i))
      ENDDO

      DO i=1, NTGRD
        DFDG(i,i)=-CONDX(I)
      ENDDO
C
C
C
      RETURN
      END
C
```

B.3 UTEMP program linked with ABAQUS stress analysis

The UTEMP program is employed in the ABAQUS thermal mechanical analysis to give the time – dependent temperature of the steel elements. Similar to the DFLUX subroutine, the "A_COEF" series contain the coefficients of time – dependent polynomial. The temperature histories of the steel elements are provided by the subroutine.

```
============================================================
      SUBROUTINE UTEMP(TEMP,NSECPT,KSTEP,KINC,TIME,NODE,COORDS)
C
      INCLUDE 'ABA_PARAM.INC'
C
      DIMENSION TEMP(NSECPT), TIME(2), COORDS(3)
      INTEGER NODE
C
C         NSECPT: Maximum number of section values required for any node in the model.
C         KSTEP: Step number.
C         KINC: Increment number.
C         TIME(1): Current value of step time.
C         TIME(2): Current value of total time.
C         NODE: Node number.
C         COORDS: An array containing the current coordinates of this point.
C                 These are the current coordinates if geometric nonlinearity is
C                  accounted for during the step (see "Procedures: overview,"
C                 Section 6.1.1); otherwise, the array contains the original
C                 coordinates of the node.
C
      INTEGER I,J,TFLAG,TAGBEAM
      REAL*8 T,X,Y,Z,POINT
      REAL*8 A1(5),A_COEF1(6,5),A_COEF2(6,5),A_COEF3(6,5),A_COEF4(6,5)
      REAL*8 A_COEF5(6,5)
      REAL*8 TTEMP

      DATA A_COEF4 /0.114922324089017E+05,   -0.544563606820375E+02,
     &    0.467942570344562E-01,   -0.125493623497931E-04,
     &    0.137717475023688E-08,   -0.539911906933580E-13,
     &   -0.266617776796129E+04,    0.125958407482212E+02,
     &   -0.107593787443667E-01,    0.287790988506668E-05,
     &   -0.315027149034340E-09,    0.123153444572136E-13,
     &    0.230671080404776E+03,   -0.108555089946981E+01,
     &    0.922920215290057E-03,   -0.246219557884539E-06,
     &    0.268799341829218E-10,   -0.104763366265212E-14,
     &   -0.879710176902699E+01,    0.411988678388759E-01,
     &   -0.348160864206935E-04,    0.925756009821290E-08,
     &   -0.100724891525802E-11,    0.391078243200157E-16,
     &    0.124893359921472E+00,   -0.581494577127306E-03,
     &    0.487787386683060E-06,   -0.129171049758698E-09,
```

```
&    0.139959820943168E-13,   -0.540868808062673E-18 /

 DATA A_COEF5 /0.131988557256828E+05,   -0.569851588648471E+02,
&    0.420749448529545E-01,   -0.105946130408785E-04,
&    0.110510825517236E-08,   -0.410501641738377E-13,
&   -0.299379822215822E+04,    0.128468957728910E+02,
&   -0.935568699189337E-02,    0.234001825704787E-05,
&   -0.242478894231168E-09,    0.893539601004941E-14,
&    0.252113681644885E+03,   -0.107373384993826E+01,
&    0.771093873285333E-03,   -0.191427573719166E-06,
&    0.196847068570964E-10,   -0.718561804018946E-15,
&   -0.932487127264960E+01,    0.393458211640789E-01,
&   -0.277629758322416E-04,    0.682614238187088E-08,
&   -0.694938994353498E-12,    0.250507100768167E-16,
&    0.128061390431244E+00,   -0.534461545967691E-03,
&    0.369205832661760E-06,   -0.896918155593795E-10,
&    0.901516798334214E-14,   -0.319673598355704E-18 /

 DATA A_COEF3 /-0.268395169528484E+06,    0.202539174611215E+04,
&   -0.262370875011493E+01,    0.761143685035437E-03,
&   -0.877515556736935E-07,    0.362250118865358E-11,
&    0.272782976724803E+05,   -0.205168468271149E+03,
&    0.265291090110622E+00,   -0.769156204137311E-04,
&    0.886335497342369E-08,   -0.365733114881228E-12,
&   -0.103769183442555E+04,    0.778119081023033E+01,
&   -0.100441670727491E-01,    0.291053952667774E-05,
&   -0.335254924023905E-09,    0.138284642069240E-13,
&    0.175133056855993E+02,   -0.130958820554099E+00,
&    0.168777964563827E-03,   -0.488844903879783E-07,
&    0.562877911546067E-11,   -0.232095404631116E-15,
&   -0.110637339657160E+00,    0.825215779314714E-03,
&   -0.106199518361489E-05,    0.307469197413533E-09,
&   -0.353923350142322E-13,    0.145893435874367E-17 /

 DATA A_COEF2 /-0.254122896512032E+06,    0.185881109971460E+04,
&   -0.235416301950590E+01,    0.673133043987084E-03,
&   -0.766748310069426E-07,    0.313241246875771E-11,
&    0.257195258015692E+05,   -0.187807080734172E+03,
&    0.237613577530055E+00,   -0.679126006543351E-04,
&    0.773312389471320E-08,   -0.315828971151399E-12,
&   -0.974694305608980E+03,    0.710645482801556E+01,
&   -0.898254788858566E-02,    0.256639261479123E-05,
&   -0.292150786675949E-09,    0.119288527453499E-13,
&    0.163948367940466E+02,   -0.119367054985162E+00,
&    0.150749310102338E-03,   -0.430581361184773E-07,
&    0.490057228310546E-11,   -0.200059066955671E-15,
&   -0.103267013579057E+00,    0.750924404628428E-03,
&   -0.947616368657343E-06,    0.270607917626069E-09,
&   -0.307940985228380E-13,    0.125696616655814E-17 /

 DATA A_COEF1 /-0.462274500424862E+05,    0.929344768945128E+03,
&   -0.161975455562515E+01,    0.485069706117613E-03,
&   -0.566559086658826E-07,    0.236639566846371E-11,
&    0.465301513707638E+04,   -0.936053073329385E+02,
&    0.163151999465583E+00,   -0.488436447587393E-04,
&    0.570327891402143E-08,   -0.238158998211759E-12,
```

```
     &   -0.176029123366810E+03,    0.353442940799141E+01,
     &   -0.615796045553729E-02,    0.184298185282143E-05,
     &   -0.215143730402964E-09,    0.898222065375870E-14,
     &    0.296916466837865E+01,   -0.593086178022304E-01,
     &    0.103239990917880E-03,   -0.308892932989239E-07,
     &    0.360514595369906E-11,   -0.150488534732096E-15,
     &   -0.188372227325999E-01,    0.373155873817366E-03,
     &   -0.648671767599675E-06,    0.194029667868069E-09,
     &   -0.226415517818101E-13,    0.944985251511202E-18 /



      T = TIME(1)
      IF(T.LT.0.0)THEN
        T = 0.0
      ENDIF

      X = COORDS(1)
      Y = COORDS(2)
      Z = COORDS(3)

      DO I=1,5
          A1(I)=0.0
      ENDDO

      TTEMP=0.0
      TFLAG=0
      TAGBEAM=0



      SELECT CASE(NODE)


      CASE(612001:612015,622001:622010)
          DO I=1,5
          DO J=1,6
              A1(I)=A1(I)+A_COEF1(J,I)*T**FLOAT(J-1)
          ENDDO
          ENDDO
          TFLAG=1
               TAGBEAM=1
        CASE(672001:672025)
          DO I=1,5
          DO J=1,6
              A1(I)=A1(I)+A_COEF2(J,I)*T**FLOAT(J-1)
          ENDDO
          ENDDO
          TFLAG=2
               TAGBEAM=1
C         CASE(722001:722025)
C         DO I=1,5
C         DO J=1,6
C                        A1(I)=A1(I)+A_COEF3(J,I)*T**FLOAT(J-1)
C         ENDDO
```

```
C         ENDDO
C         TFLAG=3
C              TAGBEAM=1
           CASE(1132001:1132010,1192001:1192010)
          DO I=1,5
          DO J=1,6
             A1(I)=A1(I)+A_COEF4(J,I)*T**FLOAT(J-1)
          ENDDO
          ENDDO
          TFLAG=4
               TAGBEAM=1
C          CASE(1142001:1142010,1202001:1202010)
C          DO I=1,5
C          DO J=1,6
C               A1(I)=A1(I)+A_COEF5(J,I)*T**FLOAT(J-1)
C          ENDDO
C          ENDDO
C          TFLAG=5
C          TAGBEAM=1


    END SELECT


    SELECT CASE(TFLAG)
    CASE (1:3)
               POINT=Y
    CASE (4:5)
               POINT=X
    END SELECT

    DO I=1,5
        TTEMP=TTEMP+A1(I)*POINT**FLOAT(I-1)
    ENDDO


                    DO I=1,NSECPT
                              TEMP(I) = TTEMP
                    ENDDO
    RETURN
    END
=================================================================
```

B.4 Program POST-FDS

The POST-FDS program is developed to generate the coefficients of the polynomial approximation for beam or column element shown in chapter 3. The FORTRAN 9.0 is used to develop this program. The coefficients are written by text file. The following is the code.

```
================================================================
C
C TO GET THE SPATIAL-TEMPORAL APPROXIMATION COEFFICIENTS OF THE COLUMNS
C AND BEAMS
C
      PROGRAM BEAM_COLUMN_APPROXIMATION_COEFFICIENTS
C
C
C
      INTEGER, PARAMETER :: FB = SELECTED_REAL_KIND(6)
      TYPE MESH_TYPE
            REAL(FB), POINTER, DIMENSION(:) :: X,Y,Z
            INTEGER :: IBAR, JBAR, KBAR, IERR
      END TYPE MESH_TYPE
      TYPE (MESH_TYPE), DIMENSION(:), ALLOCATABLE, TARGET :: MESH
      TYPE (MESH_TYPE), POINTER :: M
      INTEGER, ALLOCATABLE, DIMENSION(:) :: IOR,I1B,I2B,J1B,J2B,K1B,K2B
      INTEGER, ALLOCATABLE, DIMENSION(:,:,:) :: IT
      REAL(FB), ALLOCATABLE, DIMENSION(:,:,:) :: Q, TOUT
      CHARACTER(40) GRIDFILE, JUNK, QFILE, CHID, BNDF_FILE
      INTEGER BNDF_MESH, IJOINT
      REAL(FB) TIME
C
C
      PARAMETER(MAXMAT=1000,NTERMS=5,MAXPART=10)
      REAL*8, ALLOCATABLE, DIMENSION(:) :: CX,CY,CZ,LX,LY,LZ
      REAL*8 P4X,P4Y,P4Z,P6X,P6Y,P6Z
      REAL*8 TEMPDATA(4,MAXMAT),AVETEMP(2,MAXMAT)
      REAL*8, ALLOCATABLE, DIMENSION(:,:) :: ZONECOORD,COEF
      INTEGER, ALLOCATABLE, DIMENSION(:,:,:,:) :: ISURF
      INTEGER, ALLOCATABLE, DIMENSION(:) :: IX,IY,IZ,TIZ,NUM_ZONE
      INTEGER NOZ,L,N,NUM_LINES,CHOICEBC
      CHARACTER(40) INFO_FILE,OUT_PHRASE
      CHARACTER(40), ALLOCATABLE, DIMENSION(:) :: FILE_NAME
C
      CHARACTER(40) OUT_COEF_FILE
      INTEGER IFIRST,ILAST,IERR
C
C
      WRITE(*,*)' INPUT FILE NAME TO PROCESS(.smv file)'
C     READ(*,'(a)') CHID
    CHID = 'OaklnadRHF1'
```

```
      GRIDFILE = TRIM(CHID)//'.smv'
C
C
C
      OPEN(11, FILE=GRIDFILE, STATUS='OLD', FORM='FORMATTED')
      CALL SEARCH('NMESHES', 7, 11, IERR)
      READ(11, *) NMESHES
      ALLOCATE(MESH(NMESHES))
      REWIND(11)
      READ_SMV: DO NM = 1, NMESHES
            M=>MESH(NM)
            CALL SEARCH('GRID', 4, 11, IERR)
            READ(11, *) M%IBAR, M%JBAR, M%KBAR
            ALLOCATE(M%X(0:M%IBAR))
            ALLOCATE(M%Y(0:M%JBAR))
            ALLOCATE(M%Z(0:M%KBAR))
            CALL SEARCH('TRNX', 4, 11, IERR)
            READ(11, *) NOC
            DO I = 1, NOC
                  READ(11, *)
            ENDDO
            DO I = 0, M%IBAR
                  READ(11,*) IDUM, M%X(I)
            ENDDO
            CALL SEARCH('TRNY', 4, 11, IERR)
            READ(11, *) NOC
            DO I = 1, NOC
                  READ(11, *)
            ENDDO
            DO J = 0, M%JBAR
                  READ(11,*) IDUM, M%Y(J)
            ENDDO
            CALL SEARCH('TRNZ', 4, 11, IERR)
            READ(11, *) NOC
            DO I = 1, NOC
                  READ(11, *)
            ENDDO
            DO K = 0, M%KBAR
                  READ(11,*) IDUM, M%Z(K)
            ENDDO
      ENDDO READ_SMV
      REWIND(11)
      CALL SEARCH('BNDF', 4, 11, IERR)
      BACKSPACE(11)
      READ(11, *) JUNK, BNDF_MESH
      READ(11, '(A)') BNDF_FILE
      READ(11, '(A)')
      READ(11, *)
      READ(11, '(A)')
      CLOSE(11)
C
C
      NM = BNDF_MESH
      M=>MESH(NM)
      ALLOCATE(Q(0:M%IBAR, 0:M%JBAR, 0:M%KBAR))
      Q = 0.
```

```
      ALLOCATE(TOUT(0:M%IBAR, 0:M%JBAR, 0:M%KBAR))
      TOUT = 0.
      ALLOCATE(IT(0:M%IBAR, 0:M%JBAR, 0:M%KBAR))
      IT = 0
C
C
C
      INFO_FILE='info.dat'
      OUT_PHRASE='beamcol'
      OPEN(777,FILE=INFO_FILE,FORM='FORMATTED')
      CALL INP_BACK(777,IFIRST,ILAST,IERR)
      READ(777,*)NUM_LINES

      ALLOCATE(CX(1:NUM_LINES))
      ALLOCATE(CY(1:NUM_LINES))
      ALLOCATE(CZ(1:NUM_LINES))
      ALLOCATE(LX(1:NUM_LINES))
      ALLOCATE(LY(1:NUM_LINES))
      ALLOCATE(LZ(1:NUM_LINES))
      ALLOCATE(IX(1:NUM_LINES))
      ALLOCATE(IY(1:NUM_LINES))
      ALLOCATE(IZ(1:NUM_LINES))
      ALLOCATE(TIZ(1:NUM_LINES))
      ALLOCATE(FILE_NAME(1:NUM_LINES))
      ALLOCATE(NUM_ZONE(1:NUM_LINES))
      ALLOCATE(ZONECOORD(1:NUM_LINES,1:MAXPART))

      DO I=1,NUM_LINES
      CALL INP_BACK(777,IFIRST,ILAST,IERR)
      READ(777,*)CX(I),CY(I),CZ(I),IX(I),IY(I),IZ(I),LX(I),LY(I),LZ(I),
     &NUM_ZONE(I)

      IF(NUM_ZONE(I).NE.1)THEN
            DO J=1,NUM_ZONE(I)+1
                  CALL INP_BACK(777,IFIRST,ILAST,IERR)
                  READ(777,*)ZONECOORD(I,J)
            ENDDO
      ENDIF

      TIZ(I)=IZ(I)
      CALL MAKE_NAME(OUT_PHRASE,I,FILE_NAME(I))
      ENDDO

      CLOSE(777)

      ALLOCATE(ISURF(0:M%IBAR, 0:M%JBAR, 0:M%KBAR, 1:NUM_LINES))
      ISURF = 0

      DO L=1,NUM_LINES


      P4X=CX(L)-0.5*LX(L)
      P4Y=CY(L)-0.5*LY(L)
      P4Z=CZ(L)-0.5*LZ(L)
      P6X=CX(L)+0.5*LX(L)
      P6Y=CY(L)+0.5*LY(L)
```

```
      P6Z=CZ(L)+0.5*LZ(L)
      DO K=0,M%KBAR
      DO J=0,M%JBAR
      DO I=0,M%IBAR
            IF(((M%X(I).GT.P4X).AND.(M%X(I).LT.P6X)).AND.
     &      ((M%Y(J).GT.P4Y).AND.(M%Y(J).LT.P6Y)).AND.
     &      ((M%Z(K).GT.P4Z).AND.(M%Z(K).LT.P6Z)))THEN
                  ISURF(I,J,K,L)=1
            ENDIF
      ENDDO
      ENDDO
      ENDDO



      ENDDO
C
C
      QFILE = BNDF_FILE
      OPEN(12, FILE=QFILE, FORM='UNFORMATTED', STATUS='OLD')
      READ(12)
      READ(12)
      READ(12)
      READ(12) NPATCH

      ALLOCATE(IOR(1:NPATCH))
      ALLOCATE(I1B(1:NPATCH))
      ALLOCATE(I2B(1:NPATCH))
      ALLOCATE(J1B(1:NPATCH))
      ALLOCATE(J2B(1:NPATCH))
      ALLOCATE(K1B(1:NPATCH))
      ALLOCATE(K2B(1:NPATCH))

      DO I = 1, NPATCH
      READ(12) I1B(I), I2B(I), J1B(I), J2B(I), K1B(I), K2B(I), IOR(I)
      ENDDO
C
C
      ALLOCATE(COEF(1:MAXPART,1:NTERMS))

      OPEN(777,FILE='exacttemp',FORM='FORMATTED')

      READ_BLOOP: DO

            READ(12, END=199) TIME
            DO II = 1, NPATCH
                  READ(12, END=199)(((Q(I,J,K), I=I1B(II),I2B(II)),
     1      J=J1B(II),J2B(II)),K=K1B(II),K2B(II) )
            ENDDO


            DO II = 1, NPATCH
                  DO K = K1B(II), K2B(II)
                  DO J = J1B(II), J2B(II)
                  DO I = I1B(II), I2B(II)
                        IJOINT = 0
```

```
                                IJOINT = 1 + IT(I,J,K)
                                IF(IJOINT.EQ.1) THEN
                                        TOUT(I,J,K) = Q(I,J,K)
                                ELSE
                                        TOUT(I,J,K) = (TOUT(I,J,K)*
     1                                  FLOAT(IJOINT-1)+Q(I,J,K))/
     2                                  FLOAT(IJOINT)
                                ENDIF
                                IT(I,J,K) = IT(I,J,K) + 1
                        ENDDO
                        ENDDO
                        ENDDO
                ENDDO
C
C
        DO N=1,NUM_LINES

        OPEN(111,FILE=FILE_NAME(N),FORM='FORMATTED',POSITION='APPEND')

        DO I=1,MAXMAT
                TEMPDATA(1,I)=100000000.0
                TEMPDATA(2,I)=100000000.0
                TEMPDATA(3,I)=100000000.0
                TEMPDATA(4,I)=100000000.0
        ENDDO

        L=1
        NOZ=1

        DO K=0,M%KBAR
        DO J=0,M%JBAR
        DO I=0,M%IBAR
                IF(ISURF(I,J,K,N).NE.0)THEN
                        TEMPDATA(1,L)=M%X(I)
                        TEMPDATA(2,L)=M%Y(J)
                        TEMPDATA(3,L)=M%Z(K)
                        TEMPDATA(4,L)=TOUT(I,J,K)
                        L=L+1
                ENDIF
        ENDDO
        ENDDO
        ENDDO


        IF(IX(N).EQ.0)CHOICEBC=1
        IF(IY(N).EQ.0)CHOICEBC=2
        IF(IZ(N).EQ.0)CHOICEBC=3

        CALL MATSORT(TEMPDATA,4,MAXMAT,CHOICEBC)
        CALL ELIMISAMEPOINTS(TEMPDATA,4,MAXMAT)
        CALL CALAVETEMP(AVETEMP,TEMPDATA,4,MAXMAT,NOZ,CHOICEBC)
        CALL
REGRESSPR(NUM_ZONE(N),COEF,NTERMS,AVETEMP,MAXMAT,NOZ,MAXPART,
   &NUM_LINES,ZONECOORD,N)

        if(n.eq.10)then
```

```
          do i=1,noz
             IF(I.EQ.5)write(777,*)TIME,avetemp(2,i),
     &avetemp(1,i)
          enddo



          endif

          CALL PRINTDATATOFILE(TIME,111,COEF,NTERMS,NUM_ZONE(N),MAXPART)

          CLOSE(111)



          ENDDO
C
C
          ENDDO READ_BLOOP

          close(777)

199       CLOSE(12)
C
C
          OUT_COEF_FILE='coef_file'
          OPEN(555,FILE=OUT_COEF_FILE,FORM='FORMATTED')

          DO N=1,NUM_LINES
                CALL CAL_COL(555,FILE_NAME(N),NUM_ZONE(N),NTERMS,N)
          ENDDO

          CLOSE(555)
C
C
          STOP
          END
C
C
C---------------------------------------------------------------------
          SUBROUTINE CAL_COL(DTOPRINT,FILE_NAME,NUM_ZONE,NTERMS,N)

          PARAMETER(MAXTIME=1000)
          INTEGER DTOPRINT,NUM_ZONE,NTERMS,N
          CHARACTER(40) FILE_NAME

          REAL*8 ATIME(MAXTIME),COEF(MAXTIME,NUM_ZONE,NTERMS),A(6),
     &ACOEF(MAXTIME)
          INTEGER TIME_COUNT,I,J,K

          OPEN(111,FILE=FILE_NAME,FORM='FORMATTED')
          TIME_COUNT=0

          READ_TEMP:DO
                TIME_COUNT=TIME_COUNT+1
                READ(111,*,END=999)ATIME(TIME_COUNT)
```

```
                DO I=1,NUM_ZONE
                        DO J=1,NTERMS
                                READ(111,*)COEF(TIME_COUNT,I,J)
                        ENDDO
                ENDDO
        ENDDO READ_TEMP
999     CLOSE(111)

        TIME_COUNT=TIME_COUNT-1
C       WRITE(DTOPRINT,*)FILE_NAME

        DO I=1,NUM_ZONE
                DO J=1,NTERMS
                        DO K=1,TIME_COUNT
                                ACOEF(K)=COEF(K,I,J)
                        ENDDO
                        CALL APPROXT(ATIME,ACOEF,TIME_COUNT,6,A)

                        IF(J.EQ.1)THEN
                                IF(N.LT.10)WRITE(DTOPRINT,89)N+90,A(1),A(2)
                                IF(N.GE.10)WRITE(DTOPRINT,89)N+90,A(1),A(2)
                        ELSE
                                WRITE(DTOPRINT,88)A(1),A(2)
                        ENDIF

                        WRITE(DTOPRINT,88)A(3),A(4)

                        IF(J.EQ.NTERMS)THEN
                                WRITE(DTOPRINT,90)A(5),A(6)
                        ELSE
                                WRITE(DTOPRINT,88)A(5),A(6)
                        ENDIF

                ENDDO
        ENDDO
        WRITE(DTOPRINT,*)
88      FORMAT("     &",E25.15,",",E25.15,",")
89      FORMAT("     DATA A_COEF",I2," /",E25.15,",",E25.15,",")
90      FORMAT("     &",E25.15,",",E25.15,"/")
91      FORMAT("     DATA A_COEF",I1," /",E25.15,",",E25.15,",")
c88     FORMAT(E25.15,E25.15,E25.15,E25.15,E25.15,E25.15)


        RETURN
        END
C
C-----------------------------------------------------------------------
        SUBROUTINE SEARCH(STRING, LENGTH, LU, IERR)

        CHARACTER(*), INTENT(IN) :: STRING
        INTEGER, INTENT(OUT) :: IERR
        CHARACTER(20) :: JUNK
        INTEGER LU, LENTH

        SEARCH_LOOP: DO
        READ(LU, '(A)', END=10) JUNK
```

```
      IF (JUNK(1:LENGTH).EQ.STRING(1:LENGTH)) EXIT SEARCH_LOOP
      ENDDO SEARCH_LOOP

      IERR = 0
      RETURN

10    IERR = 1
      RETURN

      END SUBROUTINE SEARCH
C
C---------------------------------------------------------------------
    SUBROUTINE CALAVETEMP(A,B,ROWB,COLB,NOZ,CHOICEBC)

      PARAMETER(MAXMAT=1000)
      INTEGER ROWB,COLB,I,J,K,NOZ,CHOICEBC
      REAL*8 A(2,MAXMAT),ZCOORD,TEMP,B(ROWB,COLB)


      ZCOORD=B(CHOICEBC,1)
      TEMP=B(4,1)
      J=1
      K=1

      DO I=2,COLB
        IF(B(4,I).NE.100000000.0)THEN
          IF(ZCOORD.EQ.B(CHOICEBC,I))THEN
            TEMP=(TEMP*FLOAT(K)+B(4,I))/FLOAT(K+1)
            K=K+1
          ELSE
            A(1,J)=ZCOORD
            A(2,J)=TEMP
            J=J+1
            ZCOORD=B(CHOICEBC,I)
            TEMP=B(4,I)
            K=1
          ENDIF
        ENDIF

        IF(I.EQ.COLB)THEN
          A(1,J)=ZCOORD
          A(2,J)=TEMP
          NOZ=J
               ENDIF

      ENDDO

      RETURN
      END
C
C---------------------------------------------------------------------
      SUBROUTINE ELIMISAMEPOINTS(A,ROWA,COLA)

      INTEGER ROWA,COLA,I,J
      REAL*8 TEMP(ROWA),A(ROWA,COLA)
```

```
      DO I=1,COLA-1
            IF(A(4,I).NE.100000000.0)THEN
                  DO J=1,ROWA
                        TEMP(J)=A(J,I)
                  ENDDO

                  DO J=I+1,COLA
                        IF((TEMP(1).EQ.A(1,J)).AND.
     1                  (TEMP(2).EQ.A(2,J)).AND.
     2                  (TEMP(3).EQ.A(3,J)))THEN
                                    A(4,J)=0.0
                        ENDIF
                  ENDDO
            ENDIF
      ENDDO

      RETURN
      END
C
C---------------------------------------------------------------------
      SUBROUTINE MATSORT(A,ROWA,COLA,ROWN)

      INTEGER ROWA,COLA,ROWN,I,J,K
      REAL*8 TEMP,TEMP2,A(ROWA,COLA)


      DO K=1,COLA-1
            TEMP=A(ROWN,K)
            DO I=K+1, COLA
                  IF(A(ROWN,I).LT.TEMP)THEN
                        TEMP=A(ROWN,I)
                        DO J=1,ROWA
                              TEMP2=A(J,K)
                              A(J,K)=A(J,I)
                              A(J,I)=TEMP2
                        ENDDO
                  ENDIF
            ENDDO
      ENDDO

      RETURN
      END
C
C
C---------------------------------------------------------------------
      SUBROUTINE PRINTDATATOFILE(TIME,DTOPRINT,COEF,NTERMS,NUM_ZONE,
     &MAXPART)

      INTEGER, PARAMETER :: FB = SELECTED_REAL_KIND(6)
      REAL(FB)TIME
      INTEGER DTOPRINT,NTERMS,NUM_ZONE,MAXPART

      REAL*8 COEF(MAXPART,NTERMS)
      INTEGER I,J
```

```
      WRITE(DTOPRINT,*)TIME
      DO I=1,NUM_ZONE
            DO J=1,NTERMS
                  WRITE(DTOPRINT,*)COEF(I,J)
            ENDDO
      ENDDO

      RETURN
      END
C
C
C---------------------------------------------------------------------
      SUBROUTINE REGRESSPR(NUM_ZONE,COEF,NTERMS,AVETEMP,MAXMAT,NOZ,
     &MAXPART,NUM_LINES,ZONECOORD,N)

      INTEGER NUM_ZONE,NTERMS,MAXMAT,NOZ,MAXPART,NUM_LINES,N
      REAL*8 COEF(MAXPART,NTERMS),AVETEMP(2,MAXMAT),
     &ZONECOORD(NUM_LINES,MAXPART)

      INTEGER COUNT,I,J,K
      REAL*8 COLTEMP(2,MAXMAT),COEF1(NTERMS)

      DO J=1,NUM_ZONE

      IF(NUM_ZONE.NE.1)THEN

            DO K=1,MAXMAT
                  COLTEMP(1,K)=0.0
                  COLTEMP(2,K)=0.0
            ENDDO

            COUNT=0

            DO I=1,NOZ
                  IF((AVETEMP(1,I).GE.ZONECOORD(N,J)).AND.
     &                  (AVETEMP(1,I).LT.ZONECOORD(N,J+1)))THEN
                        COUNT=COUNT+1
                        COLTEMP(1,COUNT)=AVETEMP(1,I)
                        COLTEMP(2,COUNT)=AVETEMP(2,I)
                  ENDIF
            ENDDO

            CALL APPROX(COLTEMP,COUNT,NTERMS,COEF1)

      ELSE
            CALL APPROX(AVETEMP,NOZ,NTERMS,COEF1)

      ENDIF

            DO I=1,NTERMS
                  COEF(J,I)=COEF1(I)
            ENDDO
      ENDDO

      RETURN
      END
```

```
C
C-----------------------------------------------------------------------
      subroutine approx(xY,num_data,num_terms,coef)
      IMPLICIT REAL*8(A-H,O-Z)
      INTEGER NUM_DATA,NUM_TERMS
      REAL*8 xY(2,NUM_DATA),coef(NUM_TERMS),X(NUM_DATA),Y(NUM_DATA)
      REAL*8 DET
      parameter(max_data=1000, max_terms=1000)
      dimension a(num_terms+2,num_terms+2),b(num_terms+2)
      dimension a_inv(num_terms+2,num_terms+2)
      dimension base(max_terms)
      DIMENSION COEFEX(NUM_TERMS+2)
c
c
      DO I=1,NUM_DATA
            X(I)=XY(1,I)
            Y(I)=XY(2,I)
      ENDDO
      do 10 i=1,num_terms+2
      do 20 j=1,num_terms+2
      a(i,j)=0.0
20    continue
      b(i)=0.0
      COEFEX(I)=0.0
10    continue
c
c
    do 100 k=1,num_data
            call  base_functions(x(k),num_terms,base)
c
            do 30 i=1,num_terms
                  b(i)=b(i)+2.0*y(k)*base(i)
                  do 40 j=1,num_terms
                        a(i,j)=a(i,j)+2.0*base(i)*base(j)
40                  continue
30          continue
c
100   continue
c

      CALL BASE_FUNCTIONS(X(1),NUM_TERMS,BASE)
      DO I=1,NUM_TERMS
            A(I,NUM_TERMS+1)=BASE(I)
            A(NUM_TERMS+1,I)=BASE(I)
      ENDDO
      CALL BASE_FUNCTIONS(X(NUM_DATA),NUM_TERMS,BASE)
      DO I=1,NUM_TERMS
            A(I,NUM_TERMS+2)=BASE(I)
            A(NUM_TERMS+2,I)=BASE(I)
      ENDDO
      B(NUM_TERMS+1)=Y(1)
      B(NUM_TERMS+2)=Y(NUM_DATA)

      call INVERT(a,a_inv,num_terms+2,DET)
      if(det.le.0.0) then
            write(*,*) 'Error DET is less than zero'
```

```
                stop
        end if
c
    do i=1,num_terms+2
        do j=1,num_terms+2
      coefEX(i)=coefEX(i)+a_inv(i,j)*b(j)
     end do
        end do
c
C
        DO I=1,NUM_TERMS
                COEF(I)=COEFEX(I)
        ENDDO

        return
        end
C
C-----------------------------------------------------------------------
        subroutine approxt(x,y,num_data,num_terms,coef)
        implicit real*8(a-h,o-z)
    dimension x(*),y(*),coef(*)
        parameter(max_data=1000, max_terms=1000)
    dimension a(num_terms,num_terms),b(num_terms)
    dimension a_inv(num_terms,num_terms)
    dimension base(max_terms)
c
c
        do 10 i=1,num_terms
        do 20 j=1,num_terms
        a(i,j)=0.0
20      continue
        b(i)=0.0
        coef(i)=0.0
10      continue
c
c
      do 100 k=1,num_data
      call  base_functions(x(k),num_terms,base)
c
      do 30 i=1,num_terms
      b(i)=b(i)+y(k)*base(i)
      do 40 j=1,num_terms
      a(i,j)=a(i,j)+base(i)*base(j)
40      continue
30      continue
100     continue
c
      call INVERT(a,a_inv,num_terms,DET)
      if(det.le.0.0) then
        write(*,*) 'Error DET is less than zero'
        stop
        end if
c
    do i=1,num_terms
        do j=1,num_terms
      coef(i)=coef(i)+a_inv(i,j)*b(j)
```

```
       end do
            end do
c
            return
            end
C
C-----------------------------------------------------------------------
      subroutine base_functions(x,num_terms,f)
      IMPLICIT REAL*8(A-H,O-Z)
      dimension f(1000)

            DO K=1,NUM_TERMS
                  F(K)=X**( FLOAT(K-1) )
            ENDDO

            return
            end
C
C-----------------------------------------------------------------------
C CALCULATES THE INVERSE OF A MATRIX
      SUBROUTINE INVERT(A,AINV,N,DET)

      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION A(N,N),AINV(N,N),D(11,11)
      DIMENSION B(500)
            INTEGER N
            REAL*8 DET
C
      DO 20 I=1,N
      DO 10 J=1,N
10       AINV(I,J)=A(I,J)
20       B(I)=0.
C
      B(1)=1.
      CALL GAUSSJ(AINV,N,N,B,1,1,DET)
C
      IF(DET.EQ.0.) THEN
      WRITE(*,*) 'DET IS ZERO'
      END IF

      RETURN
      END
C
C-----------------------------------------------------------------------
C GAUSSIAN ELIMINATION
      SUBROUTINE gaussj(a,n,np,b,m,mp,DET)
      INTEGER m,mp,n,np,NMAX

      REAL*8 a(np,np),b(np,mp)
      PARAMETER (NMAX=100)
      INTEGER i,icol,irow,j,k,l,ll,indxc(NMAX),indxr(NMAX),ipiv(NMAX)
      REAL*8 big,dum,pivinv,DET
            REAL*8 ERRCONDITION
C           PARAMETER(ERRCONDITION=0.00000001)
C
      DET=1.0
```

```
            big=0.0
            dum=0.0
            pivinv=0.0
            LL=0
            L=0
            K=0
            ICOL=0
            IROW=0
            DO I=1,NMAX
                  INDXC(I)=0
                  INDXR(I)=0
                  IPIV(I)=0
            ENDDO
C
      do 11 j=1,n
        ipiv(j)=0
 11    continue
      do 22 i=1,n
        big=0.
        do 13 j=1,n
          if(ipiv(j).ne.1)then
            do 12 k=1,n
              if (ipiv(k).eq.0) then
                if (abs(a(j,k)).ge.big)then
                  big=abs(a(j,k))
                  irow=j
                  icol=k
                endif
              else if (ipiv(k).gt.1) then
                DET=0.
              endif
 12           continue
          endif
 13       continue
        ipiv(icol)=ipiv(icol)+1
        if (irow.ne.icol) then
          do 14 l=1,n
            dum=a(irow,l)
            a(irow,l)=a(icol,l)
            a(icol,l)=dum
 14         continue
          do 15 l=1,m
            dum=b(irow,l)
            b(irow,l)=b(icol,l)
            b(icol,l)=dum
 15         continue
        endif
        indxr(i)=irow
        indxc(i)=icol
        if (a(icol,icol).eq.0.) DET=0.
C         if (abs(a(icol,icol)).LE.ERRCONDITION) DET=0.
        pivinv=1./a(icol,icol)
        a(icol,icol)=1.
        do 16 l=1,n
          a(icol,l)=a(icol,l)*pivinv
 16       continue
```

```
      do 17 l=1,m
        b(icol,l)=b(icol,l)*pivinv
17      continue
      do 21 ll=1,n
        if(ll.ne.icol)then
          dum=a(ll,icol)
          a(ll,icol)=0.
          do 18 l=1,n
            a(ll,l)=a(ll,l)-a(icol,l)*dum
18          continue
          do 19 l=1,m
            b(ll,l)=b(ll,l)-b(icol,l)*dum
19          continue
        endif
21      continue
22    continue
      do 24 l=n,1,-1
        if(indxr(l).ne.indxc(l))then
          do 23 k=1,n
            dum=a(k,indxr(l))
            a(k,indxr(l))=a(k,indxc(l))
            a(k,indxc(l))=dum
23        continue
        endif
24    continue
      return
      END
C
C
C---------------------------------------------------------------------
C THIS SUB. ATTACHES AN INTEGER NUMBER (3 Digits) TO A CHARCTER
      subroutine MAKE_NAME(part1,n,fname)

      implicit real*8 (a-h,o-z)
      character part1*80,fname*80
      character num(10)*1

      num(1)="0"
      num(2)="1"
      num(3)="2"
      num(4)="3"
      num(5)="4"
      num(6)="5"
      num(7)="6"
      num(8)="7"
      num(9)="8"
      num(10)="9"

      a= dfloat(n)/100.
      ia=int(a)
      n2 = n - ia * 100
      b = dfloat(n2) / 10.
      ib = int (b)
      n3 = n2 - ib * 10
      ic = n3
```

```
      j=index (part1,"  ")
      fname = part1 (1:j-1)//'_'//num(ia+1)//num(ib+1)//num(ic+1)

      return
      end
C
C
C---------------------------------------------------------------------
      SUBROUTINE INP_BACK(INP,IFIRST,ILAST,IERR)
      IMPLICIT REAL*8(A-H,O-Z)
      PARAMETER (MAXC=20)
C
C      Skip input-file lines that:
C                  1) begin with 'C ' or 'c '
C                  2) first 20 characters are blank
C
      CHARACTER DUMC*80
C
      IFIRST=0
      ILAST=0
C
      IERR=1
10     READ(INP,'(A)',END=999) DUMC
      IF( DUMC (1:1) .EQ.'C '.OR. DUMC (1:1) .EQ.'c ') THEN
      GOTO 10
      END IF
C
      IF(INDEX ( DUMC (1:20), '                    ' ) .NE.0) THEN
      GOTO 10
      END IF
C
      DO I = 1,MAXC
       IF(DUMC (I:I) .NE. ' ') THEN
       IFIRST=I
       GOTO 20
       END IF
      END DO
C
 20    CONTINUE
      ILAST = IFIRST + INDEX(  DUMC (IFIRST:80), ' ') -2
      BACKSPACE (INP)
      IERR=0
999    IF(IERR.EQ.1) THEN
       WRITE(*,*) '>>>> ERROR(0)  Premature End of Input File '
      END IF
      RETURN
      END
```

B.5 Program PreFDS

The PreFDS is developed to generate the FDS model for curved bridge. The FDS accepts only a hexahedron object. Therefore, the curved objective is needed to be separated to small element to make the curved bridge model. It is developed to Visual C++ Windows based program. The following is the main function.

```
============================================================
void CprefdsDlg::OnBnClickedButton1()
{
	CFileDialog dlg(FALSE, "out", "*.out", NULL, "TEXT Type(*.out)");

	if((dlg.DoModal()) == IDOK)
	{
		FILE *out;
		out=fopen(dlg.GetPathName(),"w");

		CString str;
		double xx,yy_l,yy_u,delta_x;
		double Ax,Ay,Bx,By;
		double a,b;
		UINT i;

		xx = 30.0;
		delta_x=0.5;
		Ax=41.12;
		Ay=0.11;
		Bx=76.70;
		By=9.15;

//		for( i = 1; i < 140; i++)
		for( i = 1; i < 40; i++)
		{
// I-80
//			yy_l=-0.94*xx+56.0;
//			yy_u=yy_l+17.0;

// Fire
			yy_l=-0.94*xx+56.0+6.0;
			yy_u=yy_l+10.0;

// ground road
//			a=(Ay-By)/(Ax-Bx);
//			b=Ay-a*Ax;
//			yy_l=a*xx+b;
//			yy_u=yy_l+10.0;
```

```
//
                        fprintf(out, "&OBST XB =\t%5.2f,",xx);
                        fprintf(out, "%5.2f,",xx+delta_x);
                        fprintf(out, "\n");
                        fprintf(out, "\t\t%5.2f,",yy_l);
                        fprintf(out, "%5.2f,",yy_u);
                        fprintf(out, "\n");
                        fprintf(out, "\t\t 5.50, 6.00,");
                        fprintf(out, "\n");
                        fprintf(out, "\t\tSURF_ID = 'FIRE' /");
                        fprintf(out, "\n");
                        xx = xx + delta_x;
                }
                fclose(out);
        }
}

void CprefdsDlg::OnBnClickedButton2()
{
        CFileDialog dlg(FALSE, "out", "*.out", NULL, "TEXT Type(*.out)");

        if((dlg.DoModal()) == IDOK)
        {
                FILE *out;
                out=fopen(dlg.GetPathName(),"w");

                CString str;
                double a,b,c;
                double Ax,Ay,Bx,By,Cx,Cy;
                double xx,yy_l,yy_u,delta_x,d_road;
                UINT i;

                xx=0.0;
                delta_x=0.5;
                d_road=14.0;

                Ax=0.0;
                Ay=32.66;
                Bx=43.36;
                By=21.24;
                Cx=69.78;
                Cy=10.51;

                for( i = 1; i < 160; i++)
                {
                        a=(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))/((Ax-Bx)*(Bx-Cx)*(Cx-Ax));
                        b=(Ay-By)/(Ax-Bx)-(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))*(Ax+Bx)/((Bx-
Cx)*(Cx-Ax)*(Ax-Bx));
                        c=Ay-a*Ax*Ax-b*Ax;

                        yy_l=a*xx*xx+b*xx+c;
                        yy_u=yy_l+d_road;

// for concrete slab
/*                      fprintf(out, "&OBST XB =\t%5.2f,",xx);
                        fprintf(out, "%5.2f,",xx+delta_x);
```

```
                        fprintf(out, "\n");
                        fprintf(out, "\t\t%5.2f,",yy_l);
                        fprintf(out, "%5.2f,",yy_u);
                        fprintf(out, "\n");
                        fprintf(out, "\t\t10.00,10.50,");
                        fprintf(out, "\n");
                        fprintf(out, "\t\tSURF_ID = 'CONCRETE' /");
                        fprintf(out, "\n");
                        xx = xx + delta_x;
*/
// for steel girder
                        double d_y=yy_u - yy_l;
                        for(int j=1;j<7;j++)
                        {
                        fprintf(out, "&OBST XB =\t%5.2f,",xx);
                        fprintf(out, "%5.2f,",xx+delta_x);
                        fprintf(out, "\n");
                        double yyy = d_y * (0.05+(j-1)*0.18);
                        fprintf(out, "\t\t%5.2f,",yyy+yy_l);
                        fprintf(out, "%5.2f,",yyy+yy_l);
                        fprintf(out, "\n");
                        fprintf(out, "\t\t9.00,10.00,");
                        fprintf(out, "\n");
                        fprintf(out, "\t\tSURF_ID = 'STEEL' /");
                        fprintf(out, "\n");
                        }
                        xx = xx + delta_x;
                }
                fclose(out);
        }
}
```

B.6 Program PreABAQUS

Similar to the previous program, the PreABAQUS is developed to generate the ABAQUS FE model for curved bridge. It includes a bridge deck with shell element and steel girders with shell and beam element. It is also developed to Visual C++ Windows based program. The following is the main function.

```
============================================================
void CPreAbaqusDlg::OnBnClickedButton1()
{
	CFileDialog dlg(FALSE, "inp", "*.inp", NULL, "TEXT Type(*.inp)");

	if((dlg.DoModal()) == IDOK)
	{
		FILE *out;
		out=fopen(dlg.GetPathName(),"w");

		double a,b,c;
		double a_o,b_o,c_o;                                          // for original
concrete slab curvature
		double Ax,Ay,Bx,By,Bx2,By2,Cx,Cy;
//		double xx,yy_l,yy_u,delta_x,delta_z,d_road,zz;
		double xx,yy_l,delta_x,delta_z,zz;
		UINT i,j,k,vj,sj,vi;
		double Ax1,Ay1;
		double a1,b1;
		double d_yy;
		double d_radi;

// reivse the concrete slab mesh to fit the straight girders
		double R_a,R_b,R_c,R_fac;

		UINT nodenum,elenum, itemp;


		xx=13.0;		// starting x coordinate
		zz=10.0-1.0;	// height of girders
		delta_x=0.5;	// inc. of x direction
		delta_z=0.1;	// inc. of height of girders
//		d_road=14.0;
		nodenum=0;			// init. node number
		elenum=0;			// init. element number
		itemp=0;			// temperay UINT variable
		d_radi = 0.0;	// init. distance between girders

		R_a=0.0;			// init. revised coefficients.
		R_b=0.0;
		R_c=0.0;
		R_fac=0.0;
```

```
/*
Curve Equation "y=ax2+bx+c"
The Coefficients for the curve equation passing tree points
(Ax,Ay),(Bx,By),(Cx,Cy)
*/
                    Ax=13.0;
                    Ay=30.110559;
                    Bx=31.12513;
                    By=27.01281;
                    Bx2=33.0;
                    By2=26.552031;
                    Cx=57.0;
                    Cy=15.85;


                    a_o=a=(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))/((Ax-Bx)*(Bx-Cx)*(Cx-Ax));
                    b_o=b=(Ay-By)/(Ax-Bx)-(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))*(Ax+Bx)/((Bx-
Cx)*(Cx-Ax)*(Ax-Bx));
                    c_o=c=Ay-a*Ax*Ax-b*Ax;


/*
Print out the ABAQUS Head part and
model name
*/
                    fprintf(out, "*HEADING\n");
                    fprintf(out, "Oakland Bridge Collapse ABAQUS FE model\n");
                    fprintf(out, "**************************************************\n");
                    fprintf(out, "** S.I. units used.\n");
                    fprintf(out, "** length: meter\n");
                    fprintf(out, "** mass: kilogram\n");
                    fprintf(out, "** time: second\n");
                    fprintf(out, "** temperature: degree Celsius\n");
                    fprintf(out, "** force: Newton\n");
                    fprintf(out, "**************************************************\n");
                    fprintf(out, "**\n**\n");

/*
Create a base steel girder
*/
          // Create Nodes
                    fprintf(out, "*NODE,NSET=NWEB1\n");
                    for(j=0; j<11; j++)
                    {
                              nodenum=1000*j;
                              for(i=0; i<89; i++)
                              {

                              if(i<37)
                              {
                                        a=0.0;
                                        b=(By-Ay)/(Bx-Ax);
                                        c=Ay-b*Ax;
                              }else{
```

```
                        a=0.0;
                        b=(Cy-By2)/(Cx-Bx2);
                        c=By2-b*Bx2;
                }

                yy_l=a*xx*xx+b*xx+c;
//              yy_u=yy_l+d_road;
                if(i==0)
                {
                        Ax1=xx;
                        Ay1=yy_l;
                }
//
                nodenum=nodenum+1;
                fprintf(out, "\t%d,",nodenum);
                fprintf(out, "\t%e,",xx);
                fprintf(out, "\t%e,",yy_l);
                fprintf(out, "\t%e",zz);
                fprintf(out, "\n%");

                xx = xx + delta_x;
                }
                zz=zz+delta_z;
                xx=13.0;
        }

// Create Elements
        fprintf(out, "*ELEMENT,TYPE=S4R,ELSET=ELWEB1\n");
        for(j=0; j<10;j++)
        {
                elenum=1000*j;
                for(i=0;i<88;i++)
                {
                        elenum=elenum+1;
                        if((i==36)||(i==37)||(i==38)||(i==39))
                                fprintf(out, "** ");
                        fprintf(out, "%d,",elenum);
                        fprintf(out, "\t%d,",elenum);
                        fprintf(out, "\t%d,",elenum+1);
                        fprintf(out, "\t%d,",elenum+1+1000);
                        fprintf(out, "\t%d",elenum+1000);
                        fprintf(out, "\n%");
                }
        }



// Create Elements for bottom flange
        fprintf(out, "*ELEMENT,TYPE=B31,ELSET=ELBF1\n");
        elenum = 0;
        for(i=0;i<88;i++)
        {
                elenum=elenum+1;
                if((i==36)||(i==37)||(i==38)||(i==39))
                        fprintf(out, "** ");
                fprintf(out, "%d,",elenum+4000000);
```

```
                    fprintf(out, "\t%d,",elenum);
                    fprintf(out, "\t%d",elenum+1);
                    fprintf(out, "\n%");
                }

        // Create Elements for top flange
                fprintf(out, "*ELEMENT,TYPE=B31,ELSET=ELTF1\n");
                elenum = 0;
                for(i=0;i<88;i++)
                {
                    elenum=elenum+1;
                    if((i==36)||(i==37)||(i==38)||(i==39))
                        fprintf(out, "** ");
                    fprintf(out, "%d,",elenum+4010000);
                    fprintf(out, "\t%d,",elenum+10000);
                    fprintf(out, "\t%d",elenum+1+10000);
                    fprintf(out, "\n%");
                }




/*
Create the other STEEL GIRDERS
*/
                for(k=2; k<7;k++)
                {
                fprintf(out, "*NODE,NSET=NWEB");
                fprintf(out, "%d\n",k);
                Ax1=13.0;

                for(i=0; i<89; i++)
                {
                    if(i<37)
                    {
                        a=0.0;
                        b=(By-Ay)/(Bx-Ax);
                        c=Ay-b*Ax;
                    }else{
                        a=0.0;
                        b=(Cy-By2)/(Cx-Bx2);
                        c=By2-b*Bx2;
                    }

                    Ay1=a*Ax1*Ax1+b*Ax1+c;
                    a1=-1.0/(2.0*a_o*Ax1+b_o);
                    b1=Ay1+Ax1/(2.0*a_o*Ax1+b_o);
                    d_radi = 2.7*(k-1);                         // the distance between
girders is 2.7 meter
                    zz = 10.0 - 1.0;                        // height of the girders (meter)

                    double kk=-2.0*a1*b1*Ax1+2.0*Ax1*a1*Ay1+d_radi*d_radi*a1*a1-
a1*a1*Ax1*Ax1-Ay1*Ay1+d_radi*d_radi+2.0*b1*Ay1-b1*b1;
```

```
                xx=1.0/2.0/(a1*a1+1)*(-2.0*a1*b1+2.0*Ax1+2.0*a1*Ay1+2.0*sqrt(kk));

                for(j=0; j<11; j++)
                {
                        nodenum=100000*(k-1)+1000*j+(i+1);
                        yy_l=a1*xx+b1;
                        fprintf(out, "\t%d,",nodenum);
                        fprintf(out, "\t%e,",xx);
                        fprintf(out, "\t%e,",yy_l);
                        fprintf(out, "\t%e",zz);
                        fprintf(out, "\n%");
                        zz=zz+delta_z;
                }
                Ax1=Ax1+delta_x;
        }

        fprintf(out, "*ELEMENT,TYPE=S4R,ELSET=ELWEB");
        fprintf(out, "%d\n",k);

        for(j=0; j<10;j++)
        {
                elenum=1000*j+100000*(k-1);
                for(i=0;i<88;i++)
                {
                        if((i==36)||(i==37)||(i==38)||(i==39))
                                fprintf(out, "** ");
                        elenum=elenum+1;
                        fprintf(out, "%d,",elenum);
                        fprintf(out, "\t%d,",elenum);
                        fprintf(out, "\t%d,",elenum+1);
                        fprintf(out, "\t%d,",elenum+1+1000);
                        fprintf(out, "\t%d",elenum+1000);
                        fprintf(out, "\n%");
                }
        }

// Create Elements for bottom flange
        fprintf(out, "*ELEMENT,TYPE=B31,ELSET=ELBF");
        fprintf(out, "%d\n",k);
        elenum = 100000*(k-1);
        for(i=0;i<88;i++)
        {
                elenum=elenum+1;
                if((i==36)||(i==37)||(i==38)||(i==39))
                        fprintf(out, "** ");
                fprintf(out, "%d,",elenum+4000000);
                fprintf(out, "\t%d,",elenum);
                fprintf(out, "\t%d",elenum+1);
                fprintf(out, "\n%");
        }

// Create Elements for top flange
        fprintf(out, "*ELEMENT,TYPE=B31,ELSET=ELTF");
        fprintf(out, "%d\n",k);
        elenum = 100000*(k-1)+10000;
        for(i=0;i<88;i++)
```

```
		{
			elenum=elenum+1;
			if((i==36)||(i==37)||(i==38)||(i==39))
				fprintf(out, "** ");
			fprintf(out, "%d,",elenum+4010000-10000);
			fprintf(out, "\t%d,",elenum);
			fprintf(out, "\t%d",elenum+1);
			fprintf(out, "\n%");
		}


		}




/*
Concrete Slab shell element
*/

			zz=10.0381;                                                    // height
of the concrete slab
			Ax1=13.0;                                                      //
starting x coordinate (13.0)

			a_o=(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))/((Ax-Bx)*(Bx-Cx)*(Cx-Ax));
			b_o=(Ay-By)/(Ax-Bx)-(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))*(Ax+Bx)/((Bx-
Cx)*(Cx-Ax)*(Ax-Bx));
			c_o=Ay-a*Ax*Ax-b*Ax;

			// Nodes - Concrete Slab
			fprintf(out, "*NODE,NSET=NCONS1\n");

			for(i=0; i<89; i++)                                            // node
numbering through the length (0 - 88)
			{
				d_radi = -2.25;                                        // init. distance
from base point(Ax1) to end point of concrete.

	// -2.25 + 5 * 0.45 = 0
				for(j=0; j<39; j++)                            // 38 X 0.45 =
17.1 meter (highway width)
				{
					if(i<37)                               // 1st span
					{
						a=0.0;
						b=(By-Ay)/(Bx-Ax);
						c=Ay-b*Ax;
					}else{                                         // 2nd
span
						a=0.0;
						b=(Cy-By2)/(Cx-Bx2);
						c=By2-b*Bx2;
					}
```

```
                                        if((i==0)||(i==36)||(i==37)||(i==38)||(i==39)||(i==40)||(i==88))
                                        {
                                        a_o=a=(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-By))/((Ax-Bx)*(Bx-
Cx)*(Cx-Ax));
                                        b_o=b=(Ay-By)/(Ax-Bx)-(Ax*(By-Cy)+Bx*(Cy-Ay)+Cx*(Ay-
By))*(Ax+Bx)/((Bx-Cx)*(Cx-Ax)*(Ax-Bx));
                                        c_o=c=Ay-a*Ax*Ax-b*Ax;
                                        }

                                        R_fac=0.7;

                                        R_a=a*(1-R_fac)+a_o*R_fac;
                                        R_b=b*(1-R_fac)+b_o*R_fac;
                                        R_c=c*(1-R_fac)+c_o*R_fac;

                                        Ay1=R_a*Ax1*Ax1+R_b*Ax1+R_c;
                                        a1=-1.0/(2.0*R_a*Ax1+R_b);
                                        b1=Ay1+Ax1/(2.0*R_a*Ax1+R_b);

                                        double kk=-2.0*a1*b1*Ax1+2.0*Ax1*a1*Ay1+d_radi*d_radi*a1*a1-
a1*a1*Ax1*Ax1-Ay1*Ay1+d_radi*d_radi+2.0*b1*Ay1-b1*b1;

                                        if(j>5)
                                        xx=1.0/2.0/(a1*a1+1)*(-
2.0*a1*b1+2.0*Ax1+2.0*a1*Ay1+2.0*sqrt(kk));
                                        if(j<5)
                                        xx=1.0/2.0/(a1*a1+1)*(-2.0*a1*b1+2.0*Ax1+2.0*a1*Ay1-
2.0*sqrt(kk));
                                        if(j==5)                                        // 5 is come from
0.45 x 5 = 2.25 (init. distance)
                                        xx=Ax1;

                                        nodenum=1000000+1000*j+(i+1);
                                        yy_l=a1*xx+b1;
                                        fprintf(out, "\t%d,",nodenum);
                                        fprintf(out, "\t%e,",xx);
                                        fprintf(out, "\t%e,",yy_l);
                                        fprintf(out, "\t%e",zz);
                                        fprintf(out, "\n%");
                                        d_radi=d_radi+0.45;                             // inc. of radi. is
0.45
                                }
                                Ax1=Ax1+delta_x;
                        }

                // Shell Elements - Concrete Slab elements
                        fprintf(out, "*ELEMENT,TYPE=S4R,ELSET=ELCONS1\n");
                        for(j=0; j<38;j++)
                        {
                                elenum=1000*j+1000000;
                                for(i=0;i<88;i++)
                                {
                                        elenum=elenum+1;
                                        fprintf(out, "%d,",elenum);
                                        fprintf(out, "\t%d,",elenum);
```

```
                                        fprintf(out, "\t%d,",elenum+1);
                                        fprintf(out, "\t%d,",elenum+1+1000);
                                        fprintf(out, "\t%d",elenum+1000);
                                        fprintf(out, "\n%");
                                }
                        }
```

```
/*
Diaphragm
*/
                        UINT iii,ii,jjj;
                        UINT i_sp;
                        Ax1 = 0.0;


                        for(k=1;k<15;k++)
                        {

                        d_radi=-2.25;

                // k<8: 1st span area
                // the length between cross beam is 9 m
                // Ax1: x cordinate
                        if(k<8)
                        {
                                switch(k)
                                {
                                case 1:
                                        Ax1=13.0;
                                        break;
                                case 2:
                                        Ax1=13.5;
                                        break;
                                case 3:
                                        Ax1=17.5;
                                        break;
                                case 4:
                                        Ax1=22.0;
                                        break;
                                case 5:
                                        Ax1=26.5;
                                        break;
                                case 6:
                                        Ax1=30.5;
                                        break;
                                case 7:
                                        Ax1=31.0;
                                        break;
                                // Ax1: x coordinate of an intersection point
                                }
```

```
                a=0.0;                                          // coefficient for
the curve
                b=(By-Ay)/(Bx-Ax);                      // coefficient for the curve
                c=Ay-b*Ax;                                      // coefficient for
the curve
        }else{
                switch(k)
                {
                case 8:
                        Ax1=33.0;
                        break;
                case 9:
                        Ax1=33.5;
                        break;
                case 10:
                        Ax1=39.0;
                        break;
                case 11:
                        Ax1=45.0;
                        break;
                case 12:
                        Ax1=51.0;
                        break;
                case 13:
                        Ax1=56.5;
                        break;
                case 14:
                        Ax1=57.0;
                        break;
                }
                a=0.0;
                b=(Cy-By2)/(Cx-Bx2);
                c=By2-b*Bx2;
        }


    // Create Nodes for Diaphragms
    // k==1,7,8,14: supported cross beams

                Ay1=a*Ax1*Ax1+b*Ax1+c;                  // y coordinate of an
intersection point

                if((k==1)||(k==7)||(k==8)||(k==14))
                {
                        zz=10.0-1.5;
                        iii=16;
                        ii=15;
                        jjj=39;
                        i_sp=1;                                 // support beam:
i_sp=1
                        a1=-1.0/(2.0*a_o*Ax1+b_o);      // the support beams considering
curvature
                        b1=Ay1+Ax1/(2.0*a_o*Ax1+b_o);
                }else{
                        zz=10.0-0.5;
                        iii=6;
```

```
                    ii=5;
                    jjj=43;
                    i_sp=0;                                                 // Diaphragms:
i_sp=0
//                  based on y=ax^2+bx+c eq.
//                  calculate Ay1: y cordinate.
                    a1=-1.0/(2.0*a*Ax1+b);          // Diaphragms are perpendicular to
girders
//                  b1=Ay1+Ax1/(2.0*a*Ax1+b);
                    a1=-1.0/(2.0*a_o*Ax1+b_o);      // Diaphragms considering
curvature
                    b1=Ay1+Ax1/(2.0*a_o*Ax1+b_o);
                }


                fprintf(out, "*NODE,NSET=NSUPP");
                fprintf(out, "%d\n",k);

                for(i=0; i<iii; i++)
                {
                    nodenum=1000*i+2000000+100000*k;
                for(j=0; j<39; j++)
                {
                    double kk=-2.0*a1*b1*Ax1+2.0*Ax1*a1*Ay1+d_radi*d_radi*a1*a1-
a1*a1*Ax1*Ax1-Ay1*Ay1+d_radi*d_radi+2.0*b1*Ay1-b1*b1;
                    if(j>5)
                    xx=1.0/2.0/(a1*a1+1)*(-2.0*a1*b1+2.0*Ax1+2.0*a1*Ay1+2.0*sqrt(kk));
                    if(j<5)
                    xx=1.0/2.0/(a1*a1+1)*(-2.0*a1*b1+2.0*Ax1+2.0*a1*Ay1-2.0*sqrt(kk));
                    if(j==5)
                    xx=Ax1;

//                  d_yy=(a1*xx+b1)-(a*xx*xx+b*xx+c);
//                  yy_l=a*xx*xx+b*xx+c+d_yy;
                    yy_l=a1*xx+b1;

                    nodenum=nodenum+1;
                    fprintf(out, "\t%d,",nodenum);
                    fprintf(out, "\t%e,",xx);
                    fprintf(out, "\t%e,",yy_l);
                    fprintf(out, "\t%e",zz);
                    fprintf(out, "\n%");


        if((i_sp==0)&&((d_radi>2.69&&d_radi<2.71)||(d_radi>5.39&&d_radi<5.41)||(d_radi>8.09&&d_
radi<8.11)||(d_radi>10.79&&d_radi<10.81)))
                    {       // i_sp==0: diaphragms
                        nodenum=nodenum+1;
                        fprintf(out, "\t%d,",nodenum);
                        fprintf(out, "\t%e,",xx);
                        fprintf(out, "\t%e,",yy_l);
                        fprintf(out, "\t%e",zz);
                        fprintf(out, "\n%");
                    }
                    d_radi=d_radi+0.45;
                }
```

```
                d_radi=-2.25;
                zz=zz+delta_z;
                }


        // Create Elements for Diaphragms - Start
                fprintf(out, "*ELEMENT,TYPE=S4R,ELSET=ELSUPP");
                fprintf(out, "%d\n",k);
                for(j=0; j<ii;j++)
                {
                        elenum=1000*j+2000000+100000*k;
                        for(i=0;i<(jjj-1);i++)
                        {
                                elenum=elenum+1;
                                if(i_sp==1)
                                {
                                }else{

        if((i==0)||(i==1)||(i==2)||(i==3)||(i==4)||(i==11)||(i==18)||(i==25)||(i==32)||(i==39)||(i==40)||(i==41
))
                                        fprintf(out, "** ");
                                }
                                fprintf(out, "%d,",elenum);
                                fprintf(out, "\t%d,",elenum);
                                fprintf(out, "\t%d,",elenum+1);
                                fprintf(out, "\t%d,",elenum+1+1000);
                                fprintf(out, "\t%d",elenum+1000);
                                fprintf(out, "\n%");
                        }
                }
        // Create Elements for Diagragms - End

                }




/*
Node Names
*/
                fprintf(out,"** \n**\n");
                fprintf(out,"**
==================================================\n");
                fprintf(out,"**
==================================================\n");
                fprintf(out,"**     Node Naming\n");
                fprintf(out,"**
==================================================\n");
```

```
		fprintf(out,"** \n**\n**\n");

	// Set Nodes Name - TOP of 1st Steel Girder
		fprintf(out,"**
==========\n");
		fprintf(out,"**     Steel girders - concrete slab\n");
		fprintf(out,"**
==========\n");
		fprintf(out,"** \n**\n");

		fprintf(out, "*NSET,NSET=NWEB1T\n");
		for(i=1;i<90;i++)
		{
			fprintf(out, "%d",i+10000);
			if(i%8==0)
			{
				fprintf(out, "\n");
			}else{
				fprintf(out, ",");
			}
		}
		fprintf(out, "\n");


	// Set Nodes Name - TOP of other Steel Girders
		for(k=2; k<7;k++)
		{
		fprintf(out, "*NSET,NSET=NWEB%dT\n",k);

		for(i=1;i<90;i++)
		{
			fprintf(out, "%d",i+10000+100000*(k-1));
			if(i%8==0)
			{
				fprintf(out, "\n");
			}else{
				fprintf(out, ",");
			}
		}
		fprintf(out, "\n");
		}




	// Set Nodes Name - Concrete Slab for connectin Steel Girders
		fprintf(out,"** \n**\n");
		fprintf(out,"**
==========\n");
		fprintf(out,"**     Concrete slab - steel girders\n");
		fprintf(out,"**
==========\n");
		fprintf(out,"** \n**\n");
		for(k=1;k<7;k++)
```

```
		{
		fprintf(out, "*NSET,NSET=NCON%dB\n",k);
		for(i=1;i<90;i++)
		{
			fprintf(out, "%d",i+1000000+(6*k-2)*1000);
			if(i%8==0)
			{
				fprintf(out, "\n");
			}else{
				fprintf(out, ",");
			}
		}
		fprintf(out, "\n");
		}




	// Name for Cross beams - connecting Concrete slab
		fprintf(out,"** \n**\n");
		fprintf(out,"**
==================================================\n");
		fprintf(out,"**     Cross beams - Concrete slab\n");
		fprintf(out,"**
==================================================\n");
		fprintf(out,"** \n**\n");

		for(k=1;k<15;k++)
		{
		if((k==1)||(k==7)||(k==8)||(k==14))
		{
		fprintf(out, "*NSET,NSET=NSUPP%d",k);
		fprintf(out, "T\n");
		for(i=1;i<(jjj+1);i++)
		{
			if((i!=6)&&(i!=12)&&(i!=18)&&(i!=24)&&(i!=30)&&(i!=36))
			{
			fprintf(out, "%d",2000000+k*100000+ii*1000+i);
			if(i%8==0)
			{
				fprintf(out, "\n");
			}else{
				fprintf(out, ",");
			}
			}
		}
		fprintf(out, "\n");
		}
		}


	// Name for Concrete Slab - Cross beams
		fprintf(out,"** \n**\n");
		fprintf(out,"**
==================================================\n");
```

```
                fprintf(out,"**     Concrete slab - cross beams\n");
                fprintf(out,"**
==========================================================\n");
                fprintf(out,"** \n**\n");

                for(k=1;k<15;k++)
                {

                UINT itemp;

                if(k==1)
                        itemp=1;
                if(k==7)
                        itemp=37;
                if(k==8)
                        itemp=41;
                if(k==14)
                        itemp=89;

                if((k==1)||(k==7)||(k==8)||(k==14))
                {
                fprintf(out, "*NSET,NSET=NCONS%d",k);
                fprintf(out, "B\n");

                for(i=1;i<40;i++)
                {
                        if((i!=6)&&(i!=12)&&(i!=18)&&(i!=24)&&(i!=30)&&(i!=36))
                        {
                        fprintf(out, "%d",1000000+itemp+(i-1)*1000);
                        if(i%8==0)
                        {
                                fprintf(out, "\n");
                        }else{
                                fprintf(out, ",");
                        }
                        }

                }
                fprintf(out, "\n");
                }

        }


          // Name for Steel girders - Cross beams
                fprintf(out,"** \n**\n");
                fprintf(out,"**
==========================================================\n");
                fprintf(out,"**     Steel girders - cross beams\n");
                fprintf(out,"**
==========================================================\n");
                fprintf(out,"** \n**\n");

                /* k: number of the cross beams
                   || | | | |||| | | | ||
                   || | | | |||| | | | ||
```

```
   || | | | |||| | | | ||
   12  3  4  5  6789 10 11 12 13 14
*/
for(k=1;k<15;k++)
{


if((k==1)||(k==7)||(k==8)||(k==14))
{
        vj=12;
        sj=1;
}else{
        sj=6;
        vj=12;
}

for(i=1;i<7;i++)
{
        fprintf(out, "*NSET,NSET=NWEB%d_",i);
        fprintf(out, "%d\n",k);

        UINT i_times;
        i_times=0;

        for(j=sj;j<vj;j++)
        {
        switch(k)
        {
        case 1:
                fprintf(out, "%d",k+(j-1)*1000+100000*(i-1));
                break;
        case 2:
                fprintf(out, "%d",k+(j-1)*1000+100000*(i-1));
                break;
        case 3:
                fprintf(out, "%d",(9*(k-2)+1)+(j-1)*1000+100000*(i-1));
                break;
        case 4:
                fprintf(out, "%d",(9*(k-2)+1)+(j-1)*1000+100000*(i-1));
                break;
        case 5:
                fprintf(out, "%d",(9*(k-2)+1)+(j-1)*1000+100000*(i-1));
                break;
        case 6:
                fprintf(out, "%d",(9*(k-2))+(j-1)*1000+100000*(i-1));
                break;
        case 7:
                fprintf(out, "%d",(9*(k-3)+1)+(j-1)*1000+100000*(i-1));
                break;
        case 8:
                fprintf(out, "%d",(9*(k-4)+5)+(j-1)*1000+100000*(i-1));
                break;
        case 9:
                fprintf(out, "%d",(9*(k-5)+6)+(j-1)*1000+100000*(i-1));
                break;
        case 10:
```

```
                        fprintf(out, "%d",53+(j-1)*1000+100000*(i-1));
                        break;
                case 11:
                        fprintf(out, "%d",65+(j-1)*1000+100000*(i-1));
                        break;
                case 12:
                        fprintf(out, "%d",77+(j-1)*1000+100000*(i-1));
                        break;
                case 13:
                        fprintf(out, "%d",88+(j-1)*1000+100000*(i-1));
                        break;
                case 14:
                        fprintf(out, "%d",89+(j-1)*1000+100000*(i-1));
                        break;
                }
                i_times=i_times+1;

                if(i_times%8==0)
                {
                        fprintf(out, "\n");
                }else{
                        fprintf(out, ",");
                }

                }
                fprintf(out, "\n");
        }









        }



    // Name for Cross beams - Steel girders
                fprintf(out,"** \n**\n");
                fprintf(out,"**
==================================================\n");
                fprintf(out,"**     Cross beams - Steel girders\n");
                fprintf(out,"**
==================================================\n");
                fprintf(out,"** \n**\n");

                /* k: number of the cross beams
                   || | | | |||| | | | ||
                   || | | | |||| | | | ||
                   || | | | |||| | | | ||
                   12  3  4  5  6789 10 11 12 13 14
                */
                for(k=1;k<15;k++)
```

```
{


if((k==1)||(k==7)||(k==8)||(k==14))
{
        sj=7;
        vj=18;
        vi=7;
}else{
        sj=2;
        vj=8;
        vi=7;
}

/* i: number of the steel girder
   6 ----------------------------
   5 ----------------------------
   4 ----------------------------
   3 ----------------------------
   2 ----------------------------
   1 ----------------------------
*/
for(i=1;i<vi;i++)
{
        fprintf(out, "*NSET,NSET=NSUP%d_",i);
        fprintf(out, "%d\n",k);

        UINT i_times;
        i_times=0;

        for(j=sj;j<vj;j++)
        {

        if((i>1)&&(vj==8)){
                fprintf(out, "%d",k*100000+(j-2)*1000+(7*i-2)+2000000);
                i_times=i_times+1;
        }else{
                fprintf(out, "%d",k*100000+(j-2)*1000+(6*i-0)+2000000);
                i_times=i_times+1;
        }

        if(i_times%8==0)
        {
                fprintf(out, "\n");
        }else{
                fprintf(out, ",");
        }

        }
        fprintf(out, "\n");
}
```

```
if(sj==2){
for(i=1;i<5;i++)
{
	fprintf(out, "*NSET,NSET=NSUP%d_",i+6);
	fprintf(out, "%d\n",k);

	UINT i_times;
	i_times=0;

	for(j=sj;j<vj;j++)
	{

	fprintf(out, "%d",k*100000+(j-2)*1000+(7*i+6)+2000000);
	i_times=i_times+1;

	if(i_times%8==0)
	{
		fprintf(out, "\n");
	}else{
		fprintf(out, ",");
	}

	}
	fprintf(out, "\n");
}
}




}
```

```
// Connecting the Bridge
	fprintf(out,"** \n**\n**\n");
	fprintf(out,"**********************************************************\n");
	fprintf(out,"**     Connecting the Bridge using Equations\n");
	fprintf(out,"**********************************************************\n");
	fprintf(out,"** \n**\n");

// Steel Girders and Cross beams
	fprintf(out,"** \n**\n");
	fprintf(out,"**********************************************************\n");
	fprintf(out,"**     Connecting Steel Girders and Cross Beams\n");
	fprintf(out,"**********************************************************\n");
	fprintf(out,"** \n**\n");

	fprintf(out,"*EQUATION\n");
	fprintf(out,"** Slave Node, Master Node\n");
```

```
fprintf(out,"** If a single node, a single node\n");
fprintf(out,"** If multi nodes, can be a single node or multi nodes\n");

for(k=1;k<7;k++)
{
for(j=1;j<15;j++)
{
for(i=1;i<7;i++)
{
        fprintf(out,"2\n");
        fprintf(out,"NSUP%d_",k);
        fprintf(out,"%d,",j);
        fprintf(out,"%d,1.0,",i);
        fprintf(out,"NWEB%d_",k);
        fprintf(out,"%d,",j);
        fprintf(out,"%d,-1.0\n",i);
}
}
}



fprintf(out,"** \n**\n");

for(k=2;k<6;k++) // k: girder number
{
for(j=2;j<14;j++)
{
for(i=1;i<7;i++)
{
        if((j!=7)&&(j!=8)){
        fprintf(out,"2\n");
        fprintf(out,"NSUP%d_",k+5);
        fprintf(out,"%d,",j);
        fprintf(out,"%d,1.0,",i);
        fprintf(out,"NWEB%d_",k);
        fprintf(out,"%d,",j);
        fprintf(out,"%d,-1.0\n",i);
        }
}
}
}



// Steel Girders and Concrete Slab
        fprintf(out,"** \n**\n");
        fprintf(out,"*************************************************************\n");
        fprintf(out,"**     Connecting Steel girders and concrete slab\n");
        fprintf(out,"*************************************************************\n");
        fprintf(out,"** \n**\n");


        for(j=1;j<7;j++)
        {
        for(i=1;i<7;i++)
```

```
	{
		fprintf(out,"2\n");
		fprintf(out,"NWEB%dT,",j);
		fprintf(out,"%d,1.0,",i);
		fprintf(out,"NCON%dB,",j);
		fprintf(out,"%d,-1.0\n",i);
	}
	}



// Cross Beams and Concrete Slab
	fprintf(out,"** \n**\n");
	fprintf(out,"*************************************************************\n");
	fprintf(out,"**     Connecting Cross Beams and concrete slab\n");
	fprintf(out,"*************************************************************\n");
	fprintf(out,"** \n**\n");

	for(i=1;i<7;i++)
	{
		fprintf(out,"2\n");
		fprintf(out,"NSUPP1T,");
		fprintf(out,"%d,1.0,",i);
		fprintf(out,"NCONS1B,");
		fprintf(out,"%d,-1.0\n",i);
	}
	for(i=1;i<7;i++)
	{
		fprintf(out,"2\n");
		fprintf(out,"NSUPP7T,");
		fprintf(out,"%d,1.0,",i);
		fprintf(out,"NCONS7B,");
		fprintf(out,"%d,-1.0\n",i);
	}
	for(i=1;i<7;i++)
	{
		fprintf(out,"2\n");
		fprintf(out,"NSUPP8T,");
		fprintf(out,"%d,1.0,",i);
		fprintf(out,"NCONS8B,");
		fprintf(out,"%d,-1.0\n",i);
	}
	for(i=1;i<7;i++)
	{
		fprintf(out,"2\n");
		fprintf(out,"NSUPP14T,");
		fprintf(out,"%d,1.0,",i);
		fprintf(out,"NCONS14B,");
		fprintf(out,"%d,-1.0\n",i);
	}



// Node name for all steel
```

```
	fprintf(out,"** \n**\n**\n");
	fprintf(out,"*************************************************************\n");
	fprintf(out,"**     Node name for all steel\n");
	fprintf(out,"*************************************************************\n");
	fprintf(out,"** \n**\n");

	fprintf(out, "*NSET,NSET=NALLS\n");
	for(i=1;i<7;i++)
	{
  fprintf(out, "NWEB%d,",i);
	}
	fprintf(out, "\n");
	for(i=1;i<15;i++)
	{
  fprintf(out, "NSUPP%d",i);
		if(i%8==0)
		{
			fprintf(out, "\n");
		}else{
			fprintf(out, ",");
		}
	}
	fprintf(out,"\n");



// Node name for boundary conditions
	fprintf(out,"** \n**\n**\n");
	fprintf(out,"*************************************************************\n");
	fprintf(out,"**     Node name for boundary conditions\n");
	fprintf(out,"*************************************************************\n");
	fprintf(out,"** \n**\n");

	fprintf(out, "*NSET,NSET=NBC1\n");
	itemp=0;
	for(j=0;j<15;j++)
	{
	for(i=1;i<40;i++)
	{
		if(((i%6)!=0)||((j<5)))
		{
		fprintf(out,"%d",i+2100000+j*1000);
		itemp++;
		if(itemp%8==0)
		{
			fprintf(out, "\n");
		}else{
			fprintf(out, ",");
		}
		}
	}
	}
	fprintf(out,"\n");

	itemp=0;
	for(i=0;i<39;i++)
```

```
{
		fprintf(out,"%d",i*1000+1000001);
		itemp++;
		if(itemp%8==0)
		{
			fprintf(out, "\n");
		}else{
			fprintf(out, ",");
		}
}
fprintf(out,"\n");

itemp=0;
for(j=0;j<6;j++)
{
for(i=0;i<10;i++)
{
		fprintf(out,"%d",i*1000+1+j*100000);
		itemp++;
		if(itemp%8==0)
		{
			fprintf(out, "\n");
		}else{
			fprintf(out, ",");
		}
}
}
fprintf(out,"\n");


fprintf(out, "*NSET,NSET=NBC2\n");
itemp=0;
for(i=1;i<40;i++)
{
		fprintf(out,"%d",i+2700000);
		itemp++;
		if(itemp%8==0)
		{
			fprintf(out, "\n");
		}else{
			fprintf(out, ",");
		}
}

fprintf(out,"\n");

fprintf(out, "*NSET,NSET=NBC3\n");
itemp=0;
for(i=1;i<40;i++)
{
		fprintf(out,"%d",i+2800000);
		itemp++;
		if(itemp%8==0)
		{
			fprintf(out, "\n");
		}else{
```

```
                fprintf(out, ",");
        }
}

fprintf(out,"\n");

fprintf(out, "*NSET,NSET=NBC4\n");
itemp=0;
for(j=0;j<15;j++)
{
for(i=1;i<40;i++)
{
        if(((i%6)!=0)||((j<5)))
        {
        fprintf(out,"%d",i+3400000+j*1000);
        itemp++;
        if(itemp%8==0)
        {
                fprintf(out, "\n");
        }else{
                fprintf(out, ",");
        }
        }
}
}
fprintf(out,"\n");

itemp=0;
for(i=0;i<39;i++)
{
        fprintf(out,"%d",i*1000+1000089);
        itemp++;
        if(itemp%8==0)
        {
                fprintf(out, "\n");
        }else{
                fprintf(out, ",");
        }
}
fprintf(out,"\n");

itemp=0;
for(j=0;j<6;j++)
{
for(i=0;i<10;i++)
{
        fprintf(out,"%d",i*1000+89+j*100000);
        itemp++;
        if(itemp%8==0)
        {
                fprintf(out, "\n");
        }else{
                fprintf(out, ",");
        }
}
}
```

```
        fprintf(out,"\n");

  // Transform
  // TRANSFORM,NSET=NBC1,TYPE=R
  // Global X,Y,Z of a, Global X,Y,Z of b
  // 1,5.72505124,0,-1,0.1746709257,0
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"*TRANSFORM,NSET=NBC1,TYPE=R\n");
        fprintf(out,"** Global X,Y,Z of a, Global X,Y,Z of b\n");
        fprintf(out,"1.0,-0.06667,0.0,1.0,15.0,0.0\n**\n");
        fprintf(out,"*TRANSFORM,NSET=NBC2,TYPE=R\n");
        fprintf(out,"** Global X,Y,Z of a, Global X,Y,Z of b\n");
        fprintf(out,"1.0,-0.3253,0.0,1.0,3.074,0.0\n**\n");
        fprintf(out,"*TRANSFORM,NSET=NBC3,TYPE=R\n");
        fprintf(out,"** Global X,Y,Z of a, Global X,Y,Z of b\n");
        fprintf(out,"1.0,-0.3253,0.0,1.0,3.074,0.0\n**\n");
        fprintf(out,"*TRANSFORM,NSET=NBC4,TYPE=R\n");
        fprintf(out,"** Global X,Y,Z of a, Global X,Y,Z of b\n");
        fprintf(out,"1.0,-0.72,0.0,1.0,1.7482,0.0\n**\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"** \n**\n");



  // Node name for all Concrete
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"**     Node name for all concrete\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"** \n**\n");

        fprintf(out, "*NSET,NSET=NALLC\n");
fprintf(out, "NCONS1\n");


  // Node name for all
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"**     Node name for all\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"** \n**\n");

        fprintf(out, "*NSET,NSET=NALL\n");
fprintf(out, "NALLS,NALLC\n");



  // Element name for all steel
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"**     Element name for all steel\n");
        fprintf(out,"**********************************************************\n");
        fprintf(out,"** \n**\n");
```

```
        fprintf(out, "*ELSET,ELSET=ELALLS\n");
        for(i=1;i<7;i++)
        {
   fprintf(out, "ELWEB%d,",i);
        }
        fprintf(out, "\n");

        for(i=1;i<15;i++)
        {
   fprintf(out, "ELSUPP%d",i);
                if(i%8==0)
                {
                        fprintf(out, "\n");
                }else{
                        fprintf(out, ",");
                }
        }
        fprintf(out,"\n");

        for(i=1;i<7;i++)
        {
   fprintf(out, "ELBF%d,",i);
        }
        fprintf(out, "\n");

        for(i=1;i<7;i++)
        {
   fprintf(out, "ELTF%d,",i);
        }
        fprintf(out, "\n");


// Element name for steel girder webs
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"**************************************************************\n");
        fprintf(out,"**     Element name for steel girder web\n");
        fprintf(out,"**************************************************************\n");
        fprintf(out,"** \n**\n");

        fprintf(out, "*ELSET,ELSET=ELALLWEB\n");
        for(i=1;i<7;i++)
        {
   fprintf(out, "ELWEB%d,",i);
        }
        fprintf(out, "\n");



// Element name for steel girder Flanges
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"**************************************************************\n");
        fprintf(out,"**     Element name for steel girder flanges\n");
        fprintf(out,"**************************************************************\n");
        fprintf(out,"** \n**\n");

        fprintf(out, "*ELSET,ELSET=ELALLFL\n");
```

```
            for(i=1;i<7;i++)
            {
    fprintf(out, "ELBF%d,",i);
            }
            fprintf(out, "\n");
            for(i=1;i<7;i++)
            {
    fprintf(out, "ELTF%d,",i);
            }
            fprintf(out, "\n");



  // Element name for diaphragms
            fprintf(out,"** \n**\n**\n");
            fprintf(out,"*************************************************************\n");
            fprintf(out,"**     Element name for diaphragms\n");
            fprintf(out,"*************************************************************\n");
            fprintf(out,"** \n**\n");

            fprintf(out, "*ELSET,ELSET=ELALLDIA\n");

            for(i=2;i<15;i++)
            {
if((i!=1)&&(i!=7)&&(i!=8)&&(i!=14))
                    {
                            fprintf(out, "ELSUPP%d",i);
                            itemp=itemp++;
                            if(itemp%8==0)
                            {
                                    fprintf(out, "\n");
                            }else{
                                    fprintf(out, ",");
                            }
                    }
            }
            fprintf(out,"\n");



  // Element name for diaphragms
            fprintf(out,"** \n**\n**\n");
            fprintf(out,"*************************************************************\n");
            fprintf(out,"**     Element name for support\n");
            fprintf(out,"*************************************************************\n");
            fprintf(out,"** \n**\n");

            fprintf(out, "*ELSET,ELSET=ELALLSUP\n");
            itemp=0;

            for(i=1;i<15;i++)
            {
if((i==1)||(i==7)||(i==8)||(i==14))
                    {
                            fprintf(out, "ELSUPP%d",i);
                            itemp=itemp++;
```

```
                        if(itemp%8==0)
                        {
                                fprintf(out, "\n");
                        }else{
                                fprintf(out, ",");
                        }
                }
        }
        fprintf(out,"\n");




  // Element name for all Concrete
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"*************************************************************\n");
        fprintf(out,"**     Element name for all concrete\n");
        fprintf(out,"*************************************************************\n");
        fprintf(out,"** \n**\n");

        fprintf(out, "*ELSET,ELSET=ELALLC\n");
fprintf(out, "ELCONS1\n");


  // Element name for Weight of Parapets (DC2)
        fprintf(out,"** \n**\n**\n");
        fprintf(out,"*************************************************************\n");
        fprintf(out,"**     Element name for Weight of Parapets (DC2)\n");
        fprintf(out,"**     Parapets size = 0.74m high, 0.48m wide\n");
        fprintf(out,"*************************************************************\n");
        fprintf(out,"** \n**\n");

        fprintf(out, "*ELSET,ELSET=ELDC2\n");

        itemp=0;
        for(i=1;i<89;i++)
        {
                fprintf(out, "%d",1000000+i);
                itemp=itemp++;
                if(itemp%8==0)
                {
                        fprintf(out, "\n");
                }else{
                        fprintf(out, ",");
                }
        }
        fprintf(out,"\n");

        itemp=0;
        for(i=1;i<89;i++)
        {
                fprintf(out, "%d",1037000+i);
                itemp=itemp++;
                if(itemp%8==0)
                {
```

```
                    fprintf(out, "\n");
                }else{
                    fprintf(out, ",");
                }
            }
            fprintf(out,"\n");



    // Element name for Wearing Surface Dead Load (DW)
            fprintf(out,"** \n**\n**\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"**     Element name for Wearing Surface Dead Load (DW)\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"** \n**\n");

            fprintf(out, "*ELSET,ELSET=ELDW\n");
        fprintf(out, "ELCONS1\n");



    // Element name for all
            fprintf(out,"** \n**\n**\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"**     Element name for all\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"** \n**\n");

            fprintf(out, "*ELSET,ELSET=ELALL\n");
            fprintf(out, "ELALLS, ELCONS1\n");



    // Steel shell section FOR GIRDERS' WEB
            fprintf(out,"** \n**\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"**     Steel girders' web\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"** \n**\n");

            fprintf(out, "*SHELL
SECTION,MATERIAL=STEEL,ELSET=ELALLWEB,TEMPERATURE=1,OFFSET=SNEG\n");
            // steel thickness and integration points
            fprintf(out, "0.0254,3\n");

    // Steel shell section FOR GIRDERS' FLANGE
            fprintf(out,"** \n**\n");
            fprintf(out,"**********************************************************\n");
            fprintf(out,"**     Steel girders' FLANGE\n");
            fprintf(out,"**********************************************************\n");
```

```
		fprintf(out,"** \n**\n");

		fprintf(out, "*BEAM
SECTION,SECTION=RECT,MATERIAL=STEEL,ELSET=ELALLFL,TEMPERATURE=VALUES\n");
		// steel thickness and integration points
		fprintf(out, "0.3,0.0381\n");

	// Steel shell section FOR DIAPHRAGMS
		fprintf(out,"** \n**\n");
		fprintf(out,"**********************************************************\n");
		fprintf(out,"**     Steel DIAPHRAGMS\n");
		fprintf(out,"**********************************************************\n");
		fprintf(out,"** \n**\n");

		fprintf(out, "*SHELL
SECTION,MATERIAL=STEEL,ELSET=ELALLDIA,TEMPERATURE=1\n");
		// steel thickness and integration points
		fprintf(out, "0.0254,3\n");

	// Steel shell section FOR SUPPORTS
		fprintf(out,"** \n**\n");
		fprintf(out,"**********************************************************\n");
		fprintf(out,"**     Steel SUPPORTS\n");
		fprintf(out,"**********************************************************\n");
		fprintf(out,"** \n**\n");

		fprintf(out, "*SHELL
SECTION,MATERIAL=STEEL,ELSET=ELALLSUP,TEMPERATURE=1\n");
		// steel thickness and integration points
		fprintf(out, "0.0254,3\n");
		fprintf(out,"** \n**\n");
		fprintf(out,"**********************************************************\n");
		fprintf(out,"**MATERIAL PROPERTIES ********************** UNIT IN Pa\n");
		fprintf(out,"**********************************************************\n");
		fprintf(out,"** \n**\n");
		fprintf(out,"*MATERIAL,NAME=STEEL\n");
		fprintf(out,"******************************\n");
		fprintf(out,"*ELASTIC,TYPE=ISOTROPIC\n");
		fprintf(out,"*** Pa, Poison, Temperature\n");
		fprintf(out,"210000000000.0,0.3,  20.0\n");
		fprintf(out,"210000000000.0,0.3, 100.0\n");
		fprintf(out,"210000000000.0,0.3, 200.0\n");
		fprintf(out,"210000000000.0,0.3, 300.0\n");
		fprintf(out,"210000000000.0,0.3, 400.0\n");
		fprintf(out,"163800000000.0,0.3, 500.0\n");
		fprintf(out," 98700000000.0,0.3, 600.0\n");
		fprintf(out," 48300000000.0,0.3, 700.0\n");
		fprintf(out," 23100000000.0,0.3, 800.0\n");
		fprintf(out," 12600000000.0,0.3, 900.0\n");
		fprintf(out,"  8400000000.0,0.3,1000.0\n");
		fprintf(out,"  4200000000.0,0.3,1100.0\n");
		fprintf(out,"  4000000000.0,0.3,2000.0\n");
		fprintf(out,"*PLASTIC                \n");
		fprintf(out,"***                20 C\n");
		fprintf(out,"301000000.0 ,0.0000000 , 20.00\n");
		fprintf(out,"301000000.0 ,0.0046232 , 20.00\n");
```

```
fprintf(out,"303000000.0 ,0.0092464 , 20.00\n");
fprintf(out,"304000000.0 ,0.0136325 , 20.00\n");
fprintf(out,"306000000.0 ,0.0177815 , 20.00\n");
fprintf(out,"389000000.0 ,0.0376969 , 20.00\n");
fprintf(out,"431000000.0 ,0.138103  , 20.00\n");
fprintf(out,"435000000.0 ,5.997905  , 20.00\n");
fprintf(out,"***                100 C\n");
fprintf(out,"300000000.0  ,  0.0      , 100.0\n");
fprintf(out,"301000000.0   ,  0.0046232 , 100.0\n");
fprintf(out,"303000000.0   ,  0.0092464 , 100.0\n");
fprintf(out,"304000000.0   ,  0.013514          , 100.0\n");
fprintf(out,"306000000.0   ,  0.0180186 , 100.0\n");
fprintf(out,"389000000.0   ,  0.0376969 , 100.0\n");
fprintf(out,"431000000.0   ,  0.13834           , 100.0\n");
fprintf(out,"435000000.0   ,  5.997905  , 100.0\n");
fprintf(out,"***                200 C\n");
fprintf(out,"241000000.0  ,   0.00000000      , 200.00\n");
fprintf(out,"280000000.0    ,   0.00450466    , 200.00\n");
fprintf(out,"294000000.0    ,   0.00948349    , 200.00\n");
fprintf(out,"302000000.0    ,   0.01410670 , 200.00\n");
fprintf(out,"306000000.0    ,   0.01861130 , 200.00\n");
fprintf(out,"388000000.0    ,   0.03769690    , 200.00\n");
fprintf(out,"430000000.0    ,   0.13845900    , 200.00\n");
fprintf(out,"434000000.0    ,   5.99738100    , 200.00\n");
fprintf(out,"***                300 C\n");
fprintf(out,"183000000.0  ,   0.0         , 300.00\n");
fprintf(out,"261000000.0  ,   0.0046232       , 300.00\n");
fprintf(out,"287000000.0    ,   0.00948349    , 300.00\n");
fprintf(out,"300000000.0    ,   0.0138696     , 300.00\n");
fprintf(out,"305000000.0    ,   0.0182557     , 300.00\n");
fprintf(out,"389000000.0    ,   0.0384081     , 300.00\n");
fprintf(out,"430000000.0    ,   0.138815      , 300.00\n");
fprintf(out,"434000000.0    ,   5.9973810  , 300.00\n");
fprintf(out,"***                400 C\n");
fprintf(out,"126000000.0 ,0.00000000 , 400.00\n");
fprintf(out,"238000000.0 ,0.00509738 , 400.00\n");
fprintf(out,"278000000.0   ,0.00948349 , 400.00\n");
fprintf(out,"298000000.0   ,0.0144623  , 400.00\n");
fprintf(out,"306000000.0   ,0.0188484  , 400.00\n");
fprintf(out,"390000000.0   ,0.0384081  , 400.00\n");
fprintf(out,"431000000.0   ,0.139052   , 400.00\n");
fprintf(out,"435000000.0 ,5.996831   , 400.00\n");
fprintf(out,"***                500                 C\n");
fprintf(out,"107000000.0 ,  0.00000000 , 500.00\n");
fprintf(out,"190000000.0   ,  0.00486029 , 500.00\n");
fprintf(out,"218000000.0   ,  0.00948349 , 500.00\n");
fprintf(out,"233000000.0   ,  0.01422520 , 500.00\n");
fprintf(out,"238000000.0   ,  0.01872990 , 500.00\n");
fprintf(out,"243000000.0,  0.03840810 , 500.00\n");
fprintf(out,"269000000.0   ,  0.13893300 , 500.00\n");
fprintf(out,"273000000.0   ,  5.99771900 , 500.00\n");
fprintf(out,"***                600                 C\n");
fprintf(out," 54000000.0 ,   0.00000000 , 600.00\n");
fprintf(out,"110000000.0 ,   0.00497883 , 600.00\n");
fprintf(out,"129000000.0 ,   0.00948349 , 600.00\n");
fprintf(out,"140000000.0 ,   0.01446230 , 600.00\n");
```

```
fprintf(out,"144000000.0 ,   0.01908550 , 600.00\n");
fprintf(out,"146000000.0 ,   0.03817100 , 600.00\n");
fprintf(out,"162000000.0 ,   0.13905200 , 600.00\n");
fprintf(out,"166000000.0 ,   5.99726800 , 600.00\n");
fprintf(out,"***               700                C\n");
fprintf(out,"23155300.0 ,  0.00000000,        700.00\n");
fprintf(out,"51990300.0    ,  0.00497883,     700.00\n");
fprintf(out,"63349500.0    ,  0.00972058,     700.00\n");
fprintf(out,"69466000.0    ,  0.01481800,     700.00\n");
fprintf(out,"70776700.0    ,  0.01944120,     700.00\n");
fprintf(out,"72961200.0    ,  0.03876380,     700.00\n");
fprintf(out,"79514600.0    ,  0.13928900,     700.00\n");
fprintf(out,"82527800.0    ,  5.99688500,     700.00\n");
fprintf(out,"***               800 C\n");
fprintf(out,"15291300.0 ,  0.00000000,        800.00\n");
fprintf(out,"27524300.0 ,  0.00497883,        800.00\n");
fprintf(out,"31019400.0 ,  0.00960203,        800.00\n");
fprintf(out,"33203900.0 ,  0.01446230,        800.00\n");
fprintf(out,"34514600.0 ,  0.01896700,        800.00\n");
fprintf(out,"34951500.0 ,  0.03900080,        800.00\n");
fprintf(out,"38883500.0 ,  0.13917000,        800.00\n");
fprintf(out,"40905400.0,  5.99779000,         800.00\n");
fprintf(out,"***               900                C\n");
fprintf(out,"10922300.0  ,   0.00000000, 900.00\n");
fprintf(out,"15728200.0     ,   0.00486029, 900.00     \n");
fprintf(out,"17475700.0     ,   0.00948349, 900.00\n");
fprintf(out,"18349500.0     ,   0.01434380, 900.00\n");
fprintf(out,"18786400.0     ,   0.01908550, 900.00\n");
fprintf(out,"19660200.0     ,   0.03852670, 900.00\n");
fprintf(out,"20970900.0     ,   0.13917000, 900.00\n");
fprintf(out,"22037000.0     ,   5.99837000, 900.00\n");
fprintf(out,"** 1000 C\n");
fprintf(out," 7427180.0 ,  0.00000000,        1000.00\n");
fprintf(out,"10485400.0    ,  0.00486029,     1000.00 \n");
fprintf(out,"11359200.0    ,  0.00936494,     1000.00\n");
fprintf(out,"12669900.0    ,  0.01422520,     1000.00\n");
fprintf(out,"12669900.0    ,  0.01884840,     1000.00\n");
fprintf(out,"12669900.0    ,  0.03840810,     1000.00\n");
fprintf(out,"13543700.0    ,  0.13881500,     1000.00\n");
fprintf(out,"14642600.0,  5.99814300,         1000.00\n");
fprintf(out,"** 1100 C\n");
fprintf(out,"3753000.0, 0.0000, 1100.0\n");
fprintf(out,"5248200.0, 0.0048, 1100.0\n");
fprintf(out,"5750100.0, 0.0096, 1100.0\n");
fprintf(out,"6016800.0, 0.0143, 1100.0\n");
fprintf(out,"6120000.0, 0.0190, 1100.0\n");
fprintf(out,"6240000.0, 0.0385, 1100.0\n");
fprintf(out,"6900000.0, 0.1391, 1100.0\n");
fprintf(out,"7000000.0, 5.9999, 1100.0\n");
fprintf(out,"*******************************\n");
fprintf(out,"*EXPANSION\n");
fprintf(out,"*** m / m C\n");
fprintf(out,"1.11E-5,  20.0\n");
fprintf(out,"1.12E-5,  50.0\n");
fprintf(out,"1.14E-5, 100.0\n");
fprintf(out,"1.15E-5, 150.0\n");
```

```
                fprintf(out,"1.17E-5, 200.0\n");
                fprintf(out,"1.19E-5, 250.0\n");
                fprintf(out,"1.21E-5, 300.0\n");
                fprintf(out,"1.23E-5, 350.0\n");
                fprintf(out,"1.24E-5, 400.0\n");
                fprintf(out,"1.26E-5, 450.0\n");
                fprintf(out,"1.28E-5, 500.0\n");
                fprintf(out,"1.30E-5, 550.0\n");
                fprintf(out,"1.32E-5, 600.0\n");
                fprintf(out,"1.33E-5, 650.0\n");
                fprintf(out,"1.35E-5, 700.0\n");
                fprintf(out,"1.37E-5, 750.0\n");
                fprintf(out,"1.39E-5, 800.0\n");
                fprintf(out,"1.41E-5, 850.0\n");
                fprintf(out,"1.42E-5, 900.0\n");
                fprintf(out,"1.44E-5, 950.0\n");
                fprintf(out,"1.46E-5,1000.0\n");
                fprintf(out,"1.48E-5,1050.0\n");
                fprintf(out,"1.50E-5,1100.0\n");
                fprintf(out,"1.51E-5,1150.0\n");
                fprintf(out,"1.53E-5,1200.0\n");
                fprintf(out,"**\n");
                fprintf(out,"*DENSITY\n");
                fprintf(out,"** kilogram / meter^3\n");
                fprintf(out,"7850.0\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");

        fprintf(out,"**********************************************************************************
\n");
                fprintf(out,"** STEEL PROPERTIES
********************************************** END\n");

        fprintf(out,"**********************************************************************************
\n");




                // Concrete shell section
                fprintf(out,"** \n**\n");
                fprintf(out,"*************************************************************\n");
                fprintf(out,"**     Concrete slab\n");
                fprintf(out,"*************************************************************\n");
                fprintf(out,"** \n**\n");

                fprintf(out, "*SHELL
SECTION,MATERIAL=CON,ELSET=ELALLC,TEMPERATURE=7\n");
      // concrete thickness and integration points
                fprintf(out, "0.20,7\n");
```

```
fprintf(out,"** \n**\n");
fprintf(out,"************************************************************\n");
fprintf(out,"**MATERIAL PROPERTIES ********************** UNIT IN Pa\n");
fprintf(out,"************************************************************\n");
fprintf(out,"** \n**\n");
fprintf(out,"*MATERIAL,NAME=CON\n");
fprintf(out,"*******************************\n");
fprintf(out,"*ELASTIC\n");
fprintf(out,"** Pa, Poison, Temperature\n");
fprintf(out,"28125000000.0,0.2,  20.0\n");
fprintf(out,"19090000000.0,0.2, 100.0\n");
fprintf(out,"13333000000.0,0.2, 200.0\n");
fprintf(out," 9743000000.0,0.2, 300.0\n");
fprintf(out," 7441000000.0,0.2, 400.0\n");
fprintf(out," 4482000000.0,0.2, 500.0\n");
fprintf(out," 2500000000.0,0.2, 600.0\n");
fprintf(out," 1625000000.0,0.2, 700.0\n");
fprintf(out,"  795000000.0,0.2, 800.0\n");
fprintf(out,"  378000000.0,0.2, 900.0\n");
fprintf(out,"  200000000.0,0.2,1000.0\n");
fprintf(out,"*******************************\n");
fprintf(out,"*EXPANSION\n");
fprintf(out,"** m/mC\n");
fprintf(out,"8.63E-06, 88.00\n");
fprintf(out,"8.22E-06,147.00\n");
fprintf(out,"9.12E-06,200.00\n");
fprintf(out,"9.62E-06,257.00\n");
fprintf(out,"1.06E-05,311.00\n");
fprintf(out,"1.18E-05,360.00\n");
fprintf(out,"1.39E-05,418.00\n");
fprintf(out,"1.53E-05,474.00\n");
fprintf(out,"1.95E-05,522.00\n");
fprintf(out,"2.22E-05,563.00\n");
fprintf(out,"1.90E-05,619.00\n");
fprintf(out,"1.29E-05,693.00\n");
fprintf(out,"8.84E-06,756.00\n");
fprintf(out,"*******************************\n");
fprintf(out,"*CONCRETE DAMAGED PLASTICITY\n");
fprintf(out," 31.0\n");
fprintf(out,"*CONCRETE COMPRESSION HARDENING\n");
fprintf(out,"** 20 C\n");
fprintf(out,"45000000.0,0.0,20.0\n");
fprintf(out,"47000000.0,0.0004,20.0\n");
fprintf(out,"49000000.0,0.0009,20.0\n");
fprintf(out,"48000000.5,0.0014,20.0\n");
fprintf(out,"45000000.0,0.0044,20.0\n");
fprintf(out,"32000000.0,0.0084,20.0\n");
fprintf(out,"12000000.0,0.0134,20.0\n");
fprintf(out," 6000000.0,0.0154,20.0\n");
fprintf(out," 2000000.0,0.0174,20.0\n");
fprintf(out,"** 100 C\n");
fprintf(out,"42000000.0,0.0000,100.0\n");
fprintf(out,"44000000.0,0.0003,100.0\n");
fprintf(out,"46000000.0,0.0008,100.0\n");
fprintf(out,"47000000.0,0.0018,100.0\n");
fprintf(out,"44000000.0,0.0038,100.0\n");
```

```
fprintf(out,"35000000.0,0.0078,100.0\n");
fprintf(out,"20000000.0,0.0128,100.0\n");
fprintf(out,"14000000.0,0.0148,100.0\n");
fprintf(out," 8000000.0,0.0168,100.0\n");
fprintf(out," 3000000.0,0.0198,100.0\n");
fprintf(out,"** 200 C\n");
fprintf(out,"40000000.0,0.0000,200.0\n");
fprintf(out,"44000000.0,0.0010,200.0\n");
fprintf(out,"43000000.0,0.0030,200.0\n");
fprintf(out,"37000000.0,0.0070,200.0\n");
fprintf(out,"25000000.0,0.0120,200.0\n");
fprintf(out,"19000000.0,0.0140,200.0\n");
fprintf(out,"14000000.0,0.0160,200.0\n");
fprintf(out," 7000000.0,0.0190,200.0\n");
fprintf(out," 3000000.0,0.0220,200.0\n");
fprintf(out,"** 300 C\n");
fprintf(out,"38000000.0,0.0000,300.0\n");
fprintf(out,"41000000.0,0.0011,300.0\n");
fprintf(out,"42000000.0,0.0021,300.0\n");
fprintf(out,"37000000.0,0.0061,300.0\n");
fprintf(out,"27000000.0,0.0111,300.0\n");
fprintf(out,"23000000.0,0.0131,300.0\n");
fprintf(out,"18000000.0,0.0151,300.0\n");
fprintf(out,"12000000.0,0.0181,300.0\n");
fprintf(out," 6000000.0,0.0211,300.0\n");
fprintf(out,"** 400 C\n");
fprintf(out,"32000000.0,0.0000,400.0\n");
fprintf(out,"34000000.0,0.0007,400.0\n");
fprintf(out,"36000000.0,0.0017,400.0\n");
fprintf(out,"37000000.0,0.0037,400.0\n");
fprintf(out,"27000000.0,0.0107,400.0\n");
fprintf(out,"24000000.0,0.0127,400.0\n");
fprintf(out,"20000000.0,0.0147,400.0\n");
fprintf(out,"15000000.0,0.0177,400.0\n");
fprintf(out,"10000000.0,0.0207,400.0\n");
fprintf(out,"** 500 C\n");
fprintf(out,"26000000.0,0.0000,500.0\n");
fprintf(out,"28000000.0,0.0012,500.0\n");
fprintf(out,"29000000.0,0.0022,500.0\n");
fprintf(out,"30000000.0,0.0042,500.0\n");
fprintf(out,"24000000.0,0.0092,500.0\n");
fprintf(out,"21000000.0,0.0112,500.0\n");
fprintf(out,"19000000.0,0.0132,500.0\n");
fprintf(out,"15000000.0,0.0162,500.0\n");
fprintf(out,"11000000.0,0.0192,500.0\n");
fprintf(out,"** 600 C\n");
fprintf(out,"20000000.0,0.0000,600.0\n");
fprintf(out,"21000000.0,0.0010,600.0\n");
fprintf(out,"22000000.0,0.0020,600.0\n");
fprintf(out,"22000000.5,0.0040,600.0\n");
fprintf(out,"20000000.0,0.0070,600.0\n");
fprintf(out,"18000000.0,0.0090,600.0\n");
fprintf(out,"16000000.0,0.0110,600.0\n");
fprintf(out,"14000000.0,0.0140,600.0\n");
fprintf(out,"11000000.0,0.0170,600.0\n");
fprintf(out,"** 700 C\n");
```

```
fprintf(out,"13000000.0,0.000,700.0\n");
fprintf(out,"14000000.0,0.001,700.0\n");
fprintf(out,"14000000.5,0.002,700.0\n");
fprintf(out,"15000000.0,0.004,700.0\n");
fprintf(out,"15000000.0,0.007,700.0\n");
fprintf(out,"13000000.0,0.009,700.0\n");
fprintf(out,"12000000.0,0.011,700.0\n");
fprintf(out,"10000000.0,0.014,700.0\n");
fprintf(out," 9000000.0,0.017,700.0\n");
fprintf(out,"** 800 C\n");
fprintf(out,"7000000.0,0.0000,800.0\n");
fprintf(out,"7000000.0,0.0002,800.0\n");
fprintf(out,"7000000.5,0.0012,800.0\n");
fprintf(out,"7000000.6,0.0032,800.0\n");
fprintf(out,"7000000.7,0.0062,800.0\n");
fprintf(out,"7000000.2,0.0082,800.0\n");
fprintf(out,"7000000.0,0.0102,800.0\n");
fprintf(out,"6000000.0,0.0132,800.0\n");
fprintf(out,"5000000.0,0.0162,800.0\n");
fprintf(out,"** 900 C\n");
fprintf(out,"3600000.0,0.0000,900.0\n");
fprintf(out,"3700000.0,0.0005,900.0\n");
fprintf(out,"3800000.0,0.0015,900.0\n");
fprintf(out,"3900000.0,0.0025,900.0\n");
fprintf(out,"4000000.0,0.0055,900.0\n");
fprintf(out,"3700000.0,0.0075,900.0\n");
fprintf(out,"3400000.0,0.0095,900.0\n");
fprintf(out,"3000000.0,0.0125,900.0\n");
fprintf(out,"2800000.0,0.0155,900.0\n");
fprintf(out,"** 1000 C\n");
fprintf(out,"2000000.0,0.0000,1000.0\n");
fprintf(out,"2100000.0,0.0010,1000.0\n");
fprintf(out,"2100000.0,0.0020,1000.0\n");
fprintf(out,"2200000.0,0.0050,1000.0\n");
fprintf(out,"2100000.0,0.0070,1000.0\n");
fprintf(out,"1900000.0,0.0090,1000.0\n");
fprintf(out,"1700000.0,0.0120,1000.0\n");
fprintf(out,"1500000.0,0.0150,1000.0\n");
fprintf(out,"*CONCRETE TENSION STIFFENING, TYPE=DISPLACEMENT\n");
fprintf(out,"** 20 C\n");
fprintf(out,"2900000.0,0.000000,,20.0\n");
fprintf(out,"2650000.0,0.000080,,20.0\n");
fprintf(out,"1940000.0,0.000180,,20.0\n");
fprintf(out,"1300000.0,0.000300,,20.0\n");
fprintf(out," 870000.0,0.000400,,20.0\n");
fprintf(out," 580000.0,0.000500,,20.0\n");
fprintf(out," 390000.0,0.000600,,20.0\n");
fprintf(out," 260000.0,0.000700,,20.0\n");
fprintf(out," 176000.0,0.000800,,20.0\n");
fprintf(out," 120000.0,0.000900,,20.0\n");
fprintf(out," 100000.0,0.100000,,20.0\n");
fprintf(out,"***\n");
fprintf(out,"** 100 C\n");
fprintf(out,"2828000.0,0.000000,,100.0\n");
fprintf(out,"2518000.0,0.000114,,100.0\n");
fprintf(out,"1843000.0,0.000257,,100.0\n");
```

```
fprintf(out,"1235000.0,0.000429,,100.0\n");
fprintf(out," 827000.0,0.000572,,100.0\n");
fprintf(out," 551000.0,0.000715,,100.0\n");
fprintf(out," 371000.0,0.000858,,100.0\n");
fprintf(out," 247000.0,0.001001,,100.0\n");
fprintf(out," 167000.0,0.001144,,100.0\n");
fprintf(out," 114000.0,0.001287,,100.0\n");
fprintf(out,"  95000.0,0.143000,,100.0\n");
fprintf(out,"***\n");
fprintf(out,"** 400 C\n");
fprintf(out,"2175000.0,0.000000,,400.0\n");
fprintf(out,"1988000.0,0.000275,,400.0\n");
fprintf(out,"1455000.0,0.000584,,400.0\n");
fprintf(out," 975000.0,0.000955,,400.0\n");
fprintf(out," 653000.0,0.001264,,400.0\n");
fprintf(out," 435000.0,0.001573,,400.0\n");
fprintf(out," 293000.0,0.001882,,400.0\n");
fprintf(out," 195000.0,0.002191,,400.0\n");
fprintf(out," 132000.0,0.002500,,400.0\n");
fprintf(out,"  90000.0,0.002809,,400.0\n");
fprintf(out,"  75000.0,0.309028,,400.0\n");
fprintf(out,"***\n");
fprintf(out,"** 600 C\n");
fprintf(out,"1305000.0,0.000000,,600.0\n");
fprintf(out,"1193000.0,0.000396,,600.0\n");
fprintf(out," 873000.0,0.000896,,600.0\n");
fprintf(out," 585000.0,0.001496,,600.0\n");
fprintf(out," 392000.0,0.001996,,600.0\n");
fprintf(out," 261000.0,0.002496,,600.0\n");
fprintf(out," 176000.0,0.002996,,600.0\n");
fprintf(out," 117000.0,0.003496,,600.0\n");
fprintf(out,"  79000.0,0.003996,,600.0\n");
fprintf(out,"  54000.0,0.004496,,601.0\n");
fprintf(out,"  45000.0,0.499996,,602.0\n");
fprintf(out,"***\n");
fprintf(out,"** 800 C\n");
fprintf(out," 696000.0,0.000000,,800.0\n");
fprintf(out," 636000.0,0.000327,,800.0\n");
fprintf(out," 466000.0,0.000982,,800.0\n");
fprintf(out," 312000.0,0.001768,,800.0\n");
fprintf(out," 209000.0,0.002423,,800.0\n");
fprintf(out," 139000.0,0.003078,,800.0\n");
fprintf(out,"  94000.0,0.003733,,800.0\n");
fprintf(out,"  62000.0,0.004388,,800.0\n");
fprintf(out,"  42000.0,0.005043,,800.0\n");
fprintf(out,"  29000.0,0.005698,,801.0\n");
fprintf(out,"  24000.0,0.654803,,802.0\n");
fprintf(out,"***\n");
fprintf(out,"** 1000 C\n");
fprintf(out," 261000.0,0.000000,,1000.0\n");
fprintf(out," 239000.0,0.000198,,1000.0\n");
fprintf(out," 175000.0,0.001017,,1000.0\n");
fprintf(out," 117000.0,0.002000,,1000.0\n");
fprintf(out,"  78000.0,0.002819,,1000.0\n");
fprintf(out,"  52000.0,0.003638,,1000.0\n");
fprintf(out,"  35000.0,0.004457,,1000.0\n");
```

```
                fprintf(out,"  23000.0,0.005276,,1000.0\n");
                fprintf(out,"  16000.0,0.006095,,1000.0\n");
                fprintf(out,"  11000.0,0.006914,,1001.0\n");
                fprintf(out,"   9000.0,0.818543,,1002.0\n");
                fprintf(out,"**\n");
                fprintf(out,"*DENSITY\n");
                fprintf(out,"** kilogram / meter^3\n");
                fprintf(out,"2400.0\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");

        fprintf(out,"**********************************************************************
\n");
                fprintf(out,"** Concrete Slab ***************************************************
END\n");

        fprintf(out,"**********************************************************************
\n");

                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");

        fprintf(out,"**********************************************************************
\n");
                fprintf(out,"*NSET,NSET=PLOT\n");
                fprintf(out,"1021024,1020051,1020065,200019,200065\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");

        fprintf(out,"**********************************************************************
\n");
                fprintf(out,"**\n");
                fprintf(out,"*INITIAL CONDITIONS,TYPE=TEMPERATURE\n");
                fprintf(out,"NALLS, 20.0\n");
                fprintf(out,"NALLC, 20.0, 20.0, 20.0, 20.0, 20.0, 20.0, 20.0\n");
                fprintf(out,"*BOUNDARY\n");
                fprintf(out,"NBC1, 1,1\n");
                fprintf(out,"NBC1, 3,3\n");
                fprintf(out,"NBC2, 1,3\n");
                fprintf(out,"NBC3, 1,3\n");
                fprintf(out,"NBC4, 1,1\n");
                fprintf(out,"NBC4, 3,3\n");
                fprintf(out,"**\n");
                fprintf(out,"**\n");

        fprintf(out,"**********************************************************************
\n");
                fprintf(out,"** LOAD\n");
```

```
fprintf(out,"**********************************************************************\n");
        fprintf(out,"** DEAD LOAD - 1ST STEP\n");

fprintf(out,"**********************************************************************\n");
        fprintf(out,"*STEP,NLGEOM,INC=10000\n");
        fprintf(out,"*STATIC\n");
        fprintf(out,"1.0,1.0,1.00E-6,\n");
        fprintf(out,"*DLOAD\n");
        fprintf(out,"ELALL,GRAV,1.0,0,0,-1\n");
        fprintf(out,"ELDC2,P,-16824.64\n");
        fprintf(out,"ELDW,P,-1200.0\n");
        fprintf(out,"*CONTROLS, PARAMETERS=FIELD, FIELD=DISPLACEMENT\n");
        fprintf(out,"1.0,5.0\n");
        fprintf(out,"*CONTROLS, PARAMETERS=FIELD, FIELD=ROTATION\n");
        fprintf(out,"1.0,5.0\n");
        fprintf(out,"*END STEP\n");
        fprintf(out,"***\n");

fprintf(out,"**********************************************************************\n");
        fprintf(out,"** TEMPERATURE LOAD - 2ND STEP\n");

fprintf(out,"**********************************************************************\n");
        fprintf(out,"*STEP,NLGEOM,INC=10000\n");
        fprintf(out,"*STATIC,STABILIZE\n");
        fprintf(out,"466.0,1260.0,1.00E-10,\n");
        fprintf(out,"**100.0,1260.0,1.00E-10,\n");
        fprintf(out,"**1.0,1260.0,1.00E-10,\n");
        fprintf(out,"*TEMPERATURE,USER,FILE=OaklandHTA\n");
        fprintf(out,"NALLS,\n");
        fprintf(out,"*CONTROLS, PARAMETERS=FIELD, FIELD=DISPLACEMENT\n");
        fprintf(out,"100.0,100.0\n");
        fprintf(out,"*CONTROLS, PARAMETERS=FIELD, FIELD=ROTATION\n");
        fprintf(out,"100.0,100.0\n");
        fprintf(out,"*OUTPUT,FIELD\n");
        fprintf(out,"*NODE OUTPUT,NSET=NALL\n");
        fprintf(out,"U,\n");
        fprintf(out,"NT,\n");
        fprintf(out,"*ELEMENT OUTPUT, ELSET=ELALL\n");
        fprintf(out,"S,\n");
        fprintf(out,"E,\n");
        fprintf(out,"LE,\n");
        fprintf(out,"NE,\n");
        fprintf(out,"PEEQ,\n");
        fprintf(out,"*OUTPUT,HISTORY,FREQUENCY=1\n");
        fprintf(out,"*NODE OUTPUT,NSET=PLOT\n");
        fprintf(out,"NT,\n");
        fprintf(out,"U,\n");
        fprintf(out,"*END STEP\n");

fprintf(out,"**********************************************************************\n");

        fclose(out);
}
```

REFERENCES

[1] ACI Committee 216, Guide for Determining the Fire Endurance of Concrete Elements, American Concrete Institute Committee (ACI), Report 216R1-48, 1994.

[2] Ahmed, G. and Hurst, J. P., "Modeling pore pressure, moisture, and temperature in high strength concrete columns exposed to fire," Fire Technology, vol. 35, pp. 232-262, 1999.

[5] Ali, F. and O'Connor, D., "Structural performance of rotationally restrained steel columns in fire," Fire Safety Journal, vol. 36, pp. 679-691, 2001.

[6] Alnahhal, W. I., Chiewanichakorn, M., Aref, A. J., and Alampalli, S., "Temporal thermal behavior and damage simulations of FRP deck," Journal of Bridge Engineering, vol. 11, pp. 452-464, 2006.

[7] Alnahhal, W. I., Chiewanichakorn, M., and Aref, A. J., "Simulations of structural behaviour of fibre-reinforced polymer bridge deck under thermal effects", International Journal of Materials and Product Technology, vol. 28, pp. 122-140, 2007.

[8] Bailey, C., "Holistic behaviour of concrete buildings in fire," Proceedings of the Institution of civil Engineers, vol. 152(3), pp. 199-212, 2002.

[9] Bailey, C., "Efficient arrangement of reinforcement for membrane behaviour of composite floor slabs in fire conditions," Journal of Constructional Steel Research, vol. 59, pp. 931-949, 2003.

[10] Bailey, C., "Membrane action of slab/beam composite floor systems in fire," Engineering Structures, vol. 26, pp. 1691-1703, 2004.

[11] Branco, F. A. and Mendes, P. A., "Thermal actions for concrete bridge design," Journal of Structural Engineering, vol. 119(8), pp. 2313-2231, 1993.

[12] Cai, J., Burgess, I., and Plank, R., "A generalized steel/reinforced concrete beam-column element model for fire conditions," Engineering Structures, vol. 25, pp. 817-833, 2003.

[13] Chen, J., Young, B., and Uy, B., "Behavior of High Strength Structural Steel at Elevated Temperatures," Journal of Structural Engineering, vol. 132, pp. 1948-1954, 2006.

[14] Cheng, F., Kodur, V. K. R., and Wang, T., "Stress-strain curves for high strength concrete at elevated temperature," Journal of Materials in Civil Engineering, vol. 16(1), pp. 84-90, 2004.

[15] Dotreppe, J. C., Majkut, S., and Franssen, J. M., "Failure of a tied-arch bridge submitted to a severe localized fire," Structures and Extreme Events, IABSE Symposium, pp. 272-273, 2006.

[16] Elghazouli, A. Y. and Izzuddin, B. A., "Response of idealized composite beam-slab systems under fire conditions," Journal of Constructional Steel Research, vol. 56, pp. 199-224, 2000.

[17] Elghazouli, A. Y. and Izzuddin, B. A., "Analytical assessment of the structural performance of composite floors subject to compartment fires," Fire Safety Journal, vol. 36, pp. 769-793, 2001.

[18] Elghazouli, A. Y. and Izzuddin, B. A., "Realistic Modeling of Composite and Reinforced Concrete Floor Slabs under Extreme Loading. II: Verification and Application," Journal of Structural Engineering, vol. 130, pp. 1985-1996, 2004.

[19] ENV. Eurocode 2: Design of concrete structures - Part 1-2: General rules - Structural fire design, 1992-1-2 European pre-standard, 1995.

[20] ENV. Eurocode 3: Design of Steel Structures - Part 1-2: Fire Resistance, 1993-1-2 European pre-standard, 1995.

[21] Franssen, J. M., Cooke, G. M. E., and Latham, D. J., "Numerical Simulation of a Full Scale Fire Test on a Loaded Steel Framework," Journal of Constructional Steel Research, vol. 35, pp. 377-408, 1995.

[22] Flint, G., Usmani, A., Lamont, S., Lane, B., and Torero, J., "Structural Response of Tall Buildings to Multiple Floor Fires," Journal of Structural Engineering-ASCE, vol. 133, pp. 1719-1732, 2007.

[23] Gillie, M., Usmani, A., Rotter, M., and O'Connor, M., "Modelling of heated composite floor slabs with reference to the Cardington experiments," Fire Safety Journal, vol. 36, pp. 745-767, 2001.

[24] Gillie, M., Usmani, A. S., and Rotter, J. M., "A structural analysis of the first Cardington test," Journal of Constructional Steel Research, vol. 57, pp. 581-601, 2001.

[25] Gillie, M., Usmani, A. S., and Rotter, J. M., "A structural analysis of the Cardington British Steel Corner Test," Journal of Constructional Steel Research, vol. 58, pp. 427-442, 2002.

[26] Gillie, M., The behavior of steel-framed composite structures in fire conditions, PhD thesis, The University of Edinburgh, 2000.

[27] Gopalaratnam, V. S. and Shah, S. P., "Softening Response of Plain Concrete in Direct Tension," Journal of the American Concrete Institute, vol. 82, pp.310-323, 1985.

[28] Harmathy, T. Z., "Properties of building materials at elevated temperatures," DRP Paper No. 1080 of the Division of Building Research, 1983.

[29] Harmathy, T. Z., "Properties of building materials," in SFPE Handbook of fire protection engineering, ed. DiNenno, P. J. et. al, Section 1, Chapter 26, pp. 378-391, 1988.

[30] Hibbitt, Karlsson, and Sorensen. ABAQUS/Standard User's Manual, Version 6.4-1. HKS Inc., Pawtucket, RI, USA, 2003.

[31] Huang, Z., Burgess, I., Plank, R., and Bailey, C., "Comparison of BRE simple design method for composite floor slabs in fire with non-linear FE modeling," Fire and Materials, vol. 28, pp. 127-138, 2004.

[32] Izzuddin, B. A., Tao, X. Y., and Elghazouli, A. Y., "Realistic Modeling of Composite and Reinforced Concrete Floor Slabs under Extreme Loading. I: Analytical Method," Journal of Structural Engineering, vol. 130, pp. 1972-1984, 2004.

[33] Jones, W. W., editor, CFAST – Consolidated Model of Fire Growth and Smoke Transport (Version 6) Technical Reference Guide, NIST Special Publication 1026, National Institute of Standards and Technology (NIST), 2005.

[34] Peacock, R. D., editor, CFAST – Consolidated Model of Fire Growth and Smoke Transport (Version 6) User's Guide, NIST Special Publication 1041, National Institute of Standards and Technology (NIST), 2005.

[35] Karlsson, B. and Quintiere, J. G., Enclosure Fire Dynamics. CRC Press LLC, Boca Raton, FL, pp. 39-45, 2000.

[36] Keller, T., Tracy, C., and Hugi, E., "Fire endurance of loaded and liquid-cooled GFRP slabs for construction," Composites Part A: Applied Science and Manufacturing, vol. 37(7), pp. 1055-1067, 2006.

[37] Kim, H. S., Strength evaluation of fire damaged high strength concrete by non-destructive tests, MS Thesis, Ewha Womans University, Seoul, South Korea, 2004.

[38] Kodur, V. K. R. and Sultan, M. A., "Effect of temperature on thermal properties of high-strength concrete," Journal of Materials in Civil Engineering, vol. 15(2), pp. 101-107, 2003.

[39] Lamont, S., Usmani, A. S., and Drysdale, D. D., "Heat transfer analysis of the composite slab in the Cardington frame fire tests," Fire Safety Journal, vol. 36, pp. 815-39, 2001.

[40] Li, G. and Jiang, S., "Prediction to nonlinear behavior of steel frames subjected to fire," Fire Safety Journal, vol. 32 pp. 347-368, 1999.

[41] McGrattan, K., editor, Fire Dynamics Simulator (Version 4) Technical Reference Guide, NIST Special Publication 1018, National Institute of Standards and Technology (NIST), 2005.

[42] McGrattan, K., Forney, G. P., Floyd, J. F., Hostikka, S., and Prasad, K., Fire Dynamics Simulator (Version 3) - User's Guide, NISTIR 6784, National Institute of Standards and Technology (NIST), 2002.

[43] Mendes, P. A., Valente, J. C., and Branco, F. A., "Simulation of Ship Fire under Vasco da Gama Bridge," ACI Structural Journal, vol. 97(2), pp. 285-290, 2000.

[44] Moorty, S. and Roeder, C. W., "Temperature-dependent bridge movements," Journal of Structural Engineering, vol. 118(4), pp. 1090-1105, 1992.

[45] Najjar, S. R. and Burgess, I. W., "A nonlinear analysis for three-dimensional steel frames in fire conditions," Engineering Structures, vol. 18, pp. 77-89,1996.

[46] Neves, I. C., "The Critical Temperature of Steel Columns with Restrained Thermal Elongation," Fire Safety Journal, vol. 24, pp. 211-227, 1995.

[47] Neves, I. C., Branco, F. A., and Valente, J. C., "Effects of formwork fires in bridge construction," Concrete International, vol. 19(3), pp. 41-46, 1997.

[48] Neves, I. C., Valente, J. C., and Rodrigues, J. P., "Thermal restraint and fire resistance of columns," Fire Safety Journal, vol. 37, pp. 753-771, 2002.

[49] Newman, G. M., Robinson, J. T., and Bailey, C. G., Fire Safe design: A New Approach to Multi-Storey Steel-Framed Buildings. Steel Construction Institute, Ascot, UK, 2000.

[50] Outinen, J. and Makelainen, P., "Mechanical properties of structural steel at elevated temperatures and after cooling down," Fire and Materials, vol. 28, pp. 237-251, 2004.

[51] Rodrigues, J. P., Neves, I. C., and Valente, J. C., "Experimental research on the critical temperature of compressed steel elements with restrained thermal elongation," Fire Safety Journal, vol. 35, pp. 77-98, 2000.

[52] Lee, S. J., Strength Evaluation of Fire-damaged Normal Strength Concrete, MS Thesis, Ewha Women's University, Seoul, South Korea, 2003.

[53] Sanad, A. M., Rotter, J. M., Usmani, A. S., and O'Connor, M. A., "Composite beams in large buildings under fire - numerical modelling and structural behaiour," Fire Safety Journal, vol. 35, pp. 165-188, 2000.

[54] Shin, M. K., Strength Evaluation of Fire-damaged High Strength Concrete, MS Thesis, Ewha Women's University, Seoul, South Korea, 2004.

[55] Shin, M. K., Kim, H. S., Shin, Y. S., "Structural Behavior of Flexural Member with Normal & High Strength Concrete under High Temperature," Proceedings of Korea Institute for Structural Maintenance Inspection , vol. 7(2), pp. 157-160, 2003.

[56] Silveira, A. P., Branco, F. A., and Castanheta, M., "Statistical analysis of thermal actions for concrete bridge design," Structural Engineering International: Journal of the International Association for Bridge and Structural Engineering (IABSE), vol. 10(1), pp. 33-38, 2000.

[57] Usmani, A. S., Ed., Report MD13: BS/TEST3 ABAQUS half-floor Model using elastic shell with rotational discontinuities, Corus RD&T, The School of Civil and Environmental Engineering, The University of Edinburgh, Edinburgh, UK, 2000.

[58] Valente, J. C. and Neves, I. C., "Fire resistance of steel columns with elastically restrained axial elongation and bending," Journal of Constructional Steel Research, vol. 52, pp. 319-331, 1999.

[59] Vila Real, P. M. M., Lopes, N., Silva, L. S., and Franssen, J. M., "Lateral-torsional buckling of unrestrained steel beams under fire conditions: improvement of EC3 proposal," Computers and Structures, vol. 82, pp. 1737-1744, 2004.

[60] Wang, Y. C., "Effects of Structural Continuity on Fire Resistant Design of Steel Columns in Non-sway Multi-storey Frames," Fire Safety Journal, vol. 28, pp. 101-116, 1997.

[61] Computer models for fire and smoke, Combustion Science and Engineering, Inc., http://www.firemodelsurvey.com/pdf/JASMINE_2001.pdf (Accessed July 3, 2008).

[62] Rubini, P., SOFIE-Simulation of fires in enclosures, School of Engineering, Cranfield University, U.K, http://www.cranfield.ac.uk/ (Accessed July 3, 2008).

[63] Fire Risk Evaluation and Cost Assessment Model (FiRECAM), National Research Council (NRC), http://irc.nrc-cnrc.gc.ca/fr/frhb/firecamnew_e.html (Accessed July 3, 2008).

[64] Tanker fire destroys part of MacArthur Maze 2 freeways closed near Bay Bridge, San Francisco Chronicle, http://www.sfgate.com/cgi-bin/article.cgi?f=/c/a/2007/04/29/BAGVOPHQU46.DTL (Accessed July 3, 2008).

[65] California Interchange Collapses After Tanker Fire, FOX news Channel, http://www.foxnews.com/story/0,2933,269118,00.html (Accessed July 3, 2008).

[66] History for Oakland, CA on Sunday, April 29, 2007, The weather underground, Inc., http://www.wunderground.com/history/airport/KOAK/2007/4/29/DailyHistory.html (Accessed July 3, 2008).

[67] Maze Damage & Repair Photographs, California Department of Transportation District 4, http://www.dot.ca.gov/dist4/mazedamage/mazephotos.htm (Accessed July 3, 2008).

[68] Marzouk, H. and Chen, Z. W., "Fracture energy and tension properties of high-strength concrete," Journal of the Materials in Civil Engineering, vol. 7, pp. 108-116, 1995.

[69] Bwalya, A. C., Sultan, M. A., and Benichou, N., A Pilot Survey of Fire Loads in Canadian Homes, National Research Council Canada, Ottawa, Canada, 2004.

[70] Usmani, A. S., Ed., Report MD14: BS/TEST4 ABAQUS half-floor Model using elastic shell with rotational discontinuities, Corus RD&T, The School of Civil and Environmental Engineering, The University of Edinburgh, Edinburgh, UK, 2000.

[71] Google maps, http://maps.google.com/ (Accessed July 10, 2008).

[72] Live Search Maps, http://maps.live.com/ (Accessed July 10, 2008).

[73] Public Broadcasting Service (PBS), http://www.pbs.org/newshour/bb/science/jan-june07/overpass_05-10.html (Accessed July 3, 2008)

[74] Promat International, http://www.cafcointl.com/UserFiles/File/datasheets/DS_Mandolite%20HS3_Eng.pdf (Accessed October 2, 2008)

[75] Tanker fire destroys part of MacArthur Maze / 2 freeways closed near Bay Bridge, San Francisco Chronicle, http://www.sfgate.com/cgi-bin/object/article?f=/c/a/2007/04/29/BAGVOPHQU46.DTL&o= (Accessed July 3, 2008)